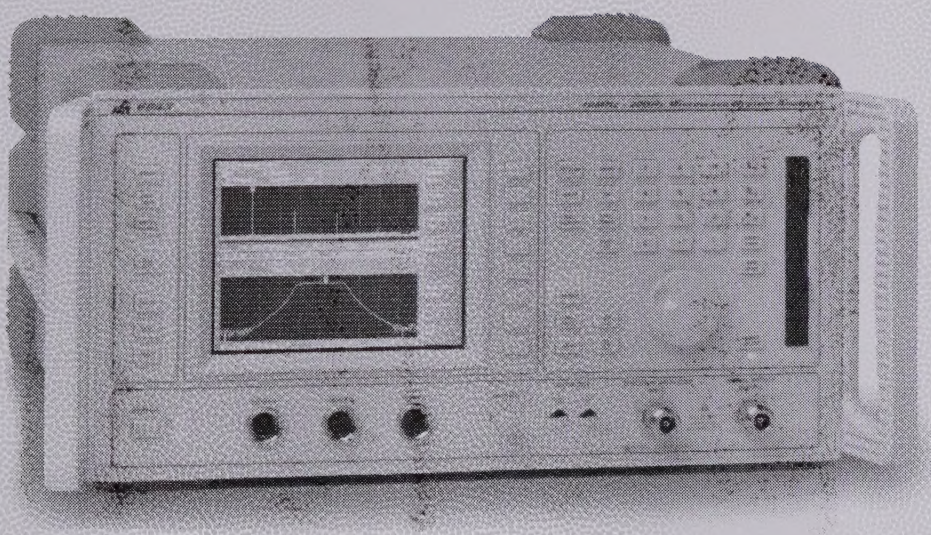
RF & MICROWAVE SCALAR & SYSTEM ANALYZERS

6820 and 6840 SERIES



Remote Operating Manual

46882/360
Issue 6

Contains pages for

Remote Operating Manual

for

RF & MICROWAVE SCALAR ANALYZERS

6820 SERIES

and

RF & MICROWAVE SYSTEM ANALYZERS

6840 SERIES

Part number 46882/360
Issue 6

Creation date 19-Jul-01

Please open and fit to the supplied Ring Binder

RF & MICROWAVE SCALAR ANALYZERS
6820 SERIES
and
RF & MICROWAVE SYSTEM ANALYZERS
6840 SERIES

© IFR Ltd. 2001

No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, or recorded by any information storage or retrieval system, without permission in writing by IFR Ltd.

Printed in the UK

Manual part no. 46882/360

Issue 6

19 July 2001

About this Manual

This manual explains how to operate the 6820 Series Scalar Analyzers and the 6840 Series System Analyzers, using either the GPIB or RS-232 interfaces.

Intended audience

Persons engaged on work relating to the design and manufacture of RF and microwave sub-systems and modules, or the installation and maintenance of these systems.

It is assumed that the reader will be familiar with the terms used in RF and microwave measurements. Although familiarity with local operation is assumed, this manual is essentially self-contained and gives all the information necessary to operate the 6800 using the GPIB or RS-232.

Structure

Chapter 1

Provides general information about the commands and conventions used when operating the instrument remotely.

Chapter 2

Provides an introduction to programming the 6800, and includes a worked example of a typical measurement.

Chapter 3

Provides specific information about the 6800 command set.

Document conventions

The following conventions apply throughout this manual:-

CAPS	Capitals are used to identify names of controls and panel markings.
[CAPS]	Capitals in square brackets indicate hard key titles.
[Italics]	Italics in square brackets indicate soft key titles.
[Averaging •]	A '•' after a soft key title indicates that the key has a toggle action, and that the function is enabled.
[Averaging o]	A 'o' after a soft key title indicates that the key has a toggle action, and that the function is disabled.

The term 6800 is used in a general sense to refer to any instrument in the 6800 series.

Associated publications

There are three other publications covering specific aspects of this equipment:-

- **Operating Manual** (46882/350). Describes local operation of the instrument, and includes performance data, installation and acceptance testing.
- **Getting Started** (46882/354). Provides example measurement procedures covering common microwave measurements.
- **Service Manual** (46880/085). Optional purchase providing information for maintenance, adjustment, calibration and repair.

Contents

		Tab numbers
Preface		
Chapter 1	GENERAL INFORMATION	1
Chapter 2	GETTING STARTED	2
Chapter 3	6800 COMMAND SET	3
Appendices		4
Appendix A	GPIB STATUS REPORTING STRUCTURE	
Appendix B	NUMERIC ENTRY HANDLING	
Appendix C	PARAMETERS NOT AFFECTED BY PRESET	
Appendix D	OVERLAPPED COMMANDS	
Appendix E	EMULATION OF IEEE 488.1 ON THE SERIAL INTERFACE	

Chapter 1
GENERAL INFORMATION

Contents

List of figures 1-1
Introduction 1-1
Background to the GPIB 1-1
SCPI 1990.0 compatibility 1-2
Compound headers 1-2
Default branches 1-3
Abbreviations 1-4
Program data 1-4
Response data 1-6
Terminating commands 1-8
Command layout 1-8
Status reporting 1-9

List of figures

Fig. 1-1 Simplified status register structure 1-10

Introduction

The instruments comprising the 6800 Series are equipped for remote operation via the GPIB interface or the RS-232 (serial) interface. The GPIB (General Purpose Interface Bus) interface provides instrument control with full talk and listen capability, and control of GPIB plotters which uses the HPGL language. The serial interface, in addition to controlling the 6800, can also be used to drive serial printers/plotters. The command syntax for both GPIB and RS-232 conforms to IEEE 488.2. The commands are common to both interfaces.

Before operating the instrument under remote control, the reader should already be familiar with making measurements with the 6800 and with the general operation of the GPIB/serial interfaces. Local operation is covered in the 6800 Operating Manual; this chapter provides a general description of the remote control commands and conventions.

Background to the GPIB

The GPIB is a high-performance bus that allows instruments and computers to be combined into integrated test systems. The bus and its associated interface operations are defined by the IEEE 488.1 standard. The later IEEE 488.2 standard defines the interface capabilities of instruments and controllers in a measurement system. This standard also defines a set of commands that a device must accept, and programming errors that a device must recognise and report.

The cables that link the devices on the bus consist of 16 signal lines which are divided into three groups:

Data bus: This comprises eight signal lines which are used to send data from one device to another. Programming commands and data sent on these lines is typically in the form of ASCII characters, although binary encoding is often used when transferring large amounts of data.

Handshake lines: The transfer of each byte of information over the data bus is controlled by a three-wire handshake process between the source of the data (talker) and all the destination device interfaces (listeners). This forces data transfers to occur at the speed of the slowest device, and ensures data integrity in multiple

listener transfers. The handshake cycle is usually performed automatically and is transparent to the GPIB programmer.

Control lines: Five control lines (or interface management lines) are used to both send bus commands and to address devices

Devices that send data over the data lines are called *talkers*; devices that receive data over the same lines are called *listeners*. *Controllers* are devices that use the control lines to specify the talker and listener in a data exchange. A GPIB system can contain more than one device with controller capabilities, but only one is allowed to control data exchanges at any given time. The device currently controlling data exchanges is called the *active controller*. One of the controller-capable devices can be designated as the *system controller*, which can take control of the bus even if it is not the active controller. Up to 15 instruments can be connected to a GPIB system.

GPIB addresses are used to identify devices on the bus. The active controller uses these addresses to specify which device talks (via a Talk Address) and which device listens (via a Listen Address) during a data exchange. Each device must therefore have a unique address, and is set on the instrument itself, using either a front-panel key sequence or a rear-panel switch. Any given device address can specify two corresponding address codes, a Talk Address and a Listen Address. The decimal equivalent of the allowable address range is 0 to 30 inclusive.

SCPI 1990.0 compatibility

Commands are divided into a number of subsystems. Each subsystem contains groups of related commands.

The form of the subsystems owes much to the ideas contained in the Standard Commands for Programmable Instruments standard, SCPI 1990.0, and the STATus subsystem and its associated Status Reporting Structures conform to that standard.

Other subsystems are instrument-specific, although some of the features of SCPI, such as the conventions for naming and organising commands, have been adopted.

Compound headers

Compound headers allow a complex set of commands to be built up from a smaller set of basic elements in a "tree" structure. The elements of a compound header are separated by a colon " : ". Each subsystem in this instrument is organised as a separate tree structure.

The use of compound headers brings a number of advantages. Commands are less cryptic compared with a traditional "flat" instrument command set, and compound header elements may appear more than once.

Example:

```
SOURce
        :FREQuency
                :CENTer\?
                :CENTre\?
                :CW\?
                :SPAN\?
                :STANdard\?
                :STARt\?
                :STOP\?
        :POWer
                :LEVel\?
                :STARt\?
                :STOP\?
```

Here the compound header elements "STARt" and "STOP" appear for both the frequency and power functions. Although it is possible to use the full compound header starting from the tree root every time, for example

SOURCE:POWER:START 2; SOURCE:POWER:STOP 10 ,

sequences of <COMMAND MESSAGE UNITS> and <QUERY MESSAGE UNITS> can often be shortened by taking advantage of the special rules which apply to compound headers. For example, having descended the tree to create the <PROGRAM MESSAGE UNIT> SOURCE:POWER:START 2, any other elements at that level may be included in the <PROGRAM MESSAGE> without repeating the entire path through the tree.

Example:

SOURCE:POWER:START 2;STOP 10

is equivalent to the two <PROGRAM MESSAGES>:

SOURCE:POWER:START 2 and SOURCE:POWER:STOP 10.

Note the use of the <PROGRAM MESSAGE UNIT SEPARATOR> character " ; " between <PROGRAM MESSAGE UNITS>.

Here is another example, this time using commands from the SYSTem subsystem (page 3-277).

Example:

SYSTEM:SERIAL:BAUD 9600;BITS 8

is equivalent to the two <PROGRAM MESSAGES>:

SYSTEM:SERIAL:BAUD 9600 and SYSTEM:SERIAL:BITS 8

To return to the top of the tree so that another "branch" may be descended, a colon is used.

Example:

SYSTEM:SERIAL:BAUD 9600;BITS 8;:SYSTEM:DATE 1998, 8, 20

Default branches

Some elements within the compound header tree structure are enclosed within square brackets, " [" and "] ". These elements may be omitted, if desired, to reduce the length of the compound header.

Example: (from the DISPlay subsystem, page 3-33)

DISPLAY:STITLE ON

is equivalent to:

DISPLAY:STITLE:STATE ON

A potential error that can occur when taking advantage of default branches is to mistake the level reached within the tree, and hence the choice of commands available at that level.

Example:

DISPLAY:STITLE ON; STRING “Filter Measurement”

will not work because STRING is not at the same level in the tree as DISPLAY:STITLE. This is because the optional [STATE] element is not used and so the level in the tree structure will remain at the level of STITLE. Refer to the following fragment of the DISPlay subsystem:

:DISPlay
:MTITle
[:STATe]
STRing
:STITle
[:STATe]
STRing

To turn on the screen title and then set the title, use:

DISPLAY:STITLE:STATE ON;STRING "Filter Measurement"

Example:

DISPLAY:STITLE ON;MTITLE ON

will work as intended since MTITLE is at the same level as DISPLAY:STITLE

Abbreviations

In general, compound header elements have a long and a short form. Following the convention adopted in SCPI, the short form of the element is printed in upper case characters, with any remaining characters in lower case. (This is merely a convenient way of showing both forms. The instrument does not distinguish between upper and lower case characters within a header).

Example:

FREQuency

The short form is "FREQ" and the long form "FREQUENCY". Other abbreviations such as "FRE" or "FREQUEN" are not allowed.

Program data

The following program data functional elements are accepted by the instrument:

<CPD> (also known as <CHARACTER PROGRAM DATA>)

<NRf> (also known as <DECIMAL NUMERIC PROGRAM DATA>)

<STRING PROGRAM DATA>

<ARBITRARY BLOCK PROGRAM DATA>

<BOOLEAN PROGRAM DATA>

All these functional elements, with the exception of <BOOLEAN PROGRAM DATA>, are defined in IEEE 488.2.

The following informal definitions are provided as a guide:

<CPD>

Character program data is used to set a parameter to one of a number of states that are best described by short alphanumeric strings.

In this manual <CPD> strings are shown using the same conventions for abbreviation as program headers; the part of the string printed in upper case characters being the short form.

Example:

INTernal, POSitive NEGative and PMETer (power meter) are the possible source levelling modes when making scalar or fault location measurements (see page 3-244). They may be abbreviated to INT, POS, NEG and PMET respectively.

<NRf>

Flexible numeric representation (also known as <DECIMAL NUMERIC PROGRAM DATA>) covers integer and floating point representations. No suffixes (e.g. kHz) are allowed

Examples:

-466	Integer value.
4.91	Explicitly placed decimal point.
59.5E+2	Mantissa and Exponent representation

The format is known as "flexible" because any of the three representations may be used for any type of numeric parameter.

Example:

Suppose a parameter requires an integer value in the range 1 to 100, and the user wishes to set its value to 42, the following values will be accepted by the instrument.

42	integer
42.0	Floating point.
4.2E1, 4200E-2	Floating point - Mantissa/exponent.
41.5	Rounded up to 42
42.4	Rounded down to 42

<NUMERIC VALUES>

This is an extension of the <NRf> format allowing suffixes and special forms as follows:

<NRf>

<NRf>+<SUFFIX PROGRAM DATA>

MINimum

MAXimum

UP

DOWN

MARKer

MINimum will set the parameter to be the nearest valid value to negative infinity.

MAXimum will set the parameter to be the nearest valid value to positive infinity.

UP will increase the value by the current step size (set using the :STEP subsystem).

DOWN will decrease the value by the current step size (set using the :STEP subsystem).

MARKer will set the parameter to the current active marker value. Either the domain value or the response will be used depending on the command issued.

An error will be generated if the parameter is meaningless for the particular command.

<STRING PROGRAM DATA>

String program data consists of a number of ASCII characters enclosed in quotes. Either a pair of single ('ASCII 39') or double ("ASCII 34") quotes may be used. If the quote character chosen to mark the beginning and end of the string also appears within it, it must be doubled.

Example:

'This string contains the word ''Hello'''

will be interpreted as the string:

'This string contains the word 'Hello'

When receiving string data, the 6800 GPIB system interprets character codes as follows:

32 - 126	Standard ASCII characters
127	Copyright symbol ©
128	Mu symbol µ
129	Degree symbol °
130	Ohms symbol Ω

Any command received with string data containing any other code will result in an error message being displayed.

<ARBITRARY BLOCK PROGRAM DATA>

This format is used to send large quantities of 8-bit binary data to the instrument.

Since it is not intended that the user should ever need to compile data of this type for transmission to the instrument, details of the format are not given here.

Note that data received from the instrument as <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA> is already in a form suitable for transmission back to the instrument as <ARBITRARY BLOCK PROGRAM DATA>. The definite length form can be used on both GPIB and RS-232 interfaces.

If the indefinite length form is used, the data must be terminated by line feed with EOI asserted. This means that a command requiring <ARBITRARY BLOCK PROGRAM DATA> must be the last <PROGRAM MESSAGE UNIT> of the <PROGRAM MESSAGE>. The indefinite length form of data cannot be used with the RS-232 interface.

<BOOLEAN PROGRAM DATA>

This is not defined in IEEE 488.2, but is a useful addition for programming parameters that have an "ON/OFF" function.

A parameter accepting <BOOLEAN PROGRAM DATA> will take the values OFF | ON | NRf value. If an NRf value is entered it is rounded to an integer, and any value other than zero is treated as ON.

Response data

The following response data functional elements are generated by the instrument:

<CRD> (also known as <CHARACTER RESPONSE DATA>)

<NR1>

<NR2>

<NR3>

<STRING RESPONSE DATA>

<DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

<ARBITRARY ASCII RESPONSE DATA>

<BOOLEAN RESPONSE DATA>

<HEXADECIMAL NUMERIC RESPONSE DATA>

All these functional elements, with the exception of <BOOLEAN RESPONSE DATA>, are defined in IEEE 488.2.

The following informal definitions are provided as a guide:

<CRD>

This type of response is returned when reading the value of a parameter which can take a number of discrete states. States are represented by short alphanumeric strings.

Example:

> INT, POS, NEG and PMET are the possible <CRD> responses if the parameter which determines the levelling mode for a scalar or fault location measurement is queried (see page 3-244).

Note that when setting the parameter, the long form (i.e. INTERNAL, POSITIVE, NEGATIVE or PMETER) may be used, but when the parameter is queried, the short form is always returned.

<NR1>

This type of numeric response is used when returning the value of integer parameters, such as averaging number or number of measurement points. A negative integer will be preceded with a – sign; a positive integer may or may not have a sign.

Examples:

15
+3
-57

<NR2>

This type of numeric response is used to return real numbers; it includes an explicitly placed decimal point, but no exponent. The decimal point will always be output, with at least one digit before and one digit after the decimal point. A negative number will be preceded with a – sign; a positive number may or may not have a + sign.

Suffixes are not output, and all values will be in the fundamental units (e.g. for frequency the value will always be in Hz, not kHz or MHz, etc.).

Examples:

17.91
-18.27
+18.83

<NR3>

This type of numeric response is used to return real numbers; it includes an explicitly placed decimal point and an exponent. The decimal point will always be output, with at least one digit before and one digit after the decimal point. A negative number will be preceded with a – sign; a positive number may or may not have a + sign. At least one digit will follow the exponent, and the sign of the exponent is always output.

Suffixes are not output, and all values will be in the fundamental units (e.g. for frequency the value will always be in Hz, not kHz or MHz, etc.).

Examples:

1.756E+2
182.8E-3

<STRING RESPONSE DATA>

This takes a similar form to <STRING PROGRAM DATA> except that the delimiting character is always a double quote, ("ASCII 34").

<DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

This form of response is used when reading blocks of 8-bit binary data from the instrument. Examples include settings stores and trace memories.

The format comprises a '#' character followed by a single digit, followed by the data length, followed by the actual data. The single digit indicates how many digits follow to specify the length.

<ARBITRARY ASCII RESPONSE DATA>

This takes the form of an ASCII string terminated by newline (ASCII 10) with EOI asserted.

Notes on interpreting data returned in this format will be found in the descriptions for the few commands that use it.

Because EOI is always used as a terminator, a <QUERY MESSAGE UNIT> which generates data in this form must be the last <QUERY MESSAGE UNIT> in the <PROGRAM MESSAGE>.

<BOOLEAN RESPONSE DATA>

This is not defined in IEEE 488.2, but is a useful addition for querying parameters that have an "ON/OFF" function.

The response is either "0" or "1", where "0" means "OFF" and "1" means "ON".

<HEXADECIMAL NUMERIC RESPONSE DATA>

This type of numeric response is used to return an integer number in hexadecimal notation.

Terminating commands

GPIB

A **<PROGRAM MESSAGE TERMINATOR>** (as defined in IEEE 488.2) can be a line-feed character (ASCII 10), a line-feed character with the ^END message asserted at the same time, or an ^END message asserted with the final character of the <PROGRAM MESSAGE>. The terminator may be preceded by any number of "white space" characters, i.e. any single ACSII-encoded byte in the range 0 to 9 and 11 to 32 decimal.

A **<RESPONSE MESSAGE TERMINATOR>** (as defined in IEEE 488.2) is a line-feed character with the ^END message asserted at the same time.

Many GPIB controllers terminate program messages with a line-feed character and, by default, accept newline as the response message terminator. When transferring binary data - which may contain embedded line-feed characters - it is necessary to ensure that the controller uses only ^END messages. Usually this requires the controller's GPIB interface to be set up to generate and detect ^END. Refer to the documentation supplied with the controller.

RS-232

For RS-232 operation the command is terminated by a line-feed character only.

Command layout

Each command is set out as follows:

1. Path from the subsystem root.

Example:

```
:SYSTem
      :ISETtings
            :KEYBoard
```

2. Parameters

The first line lists each parameter, stating its <PROGRAM DATA> functional element (as defined in IEEE 488.2).

Subsequent lines explain the meaning of each parameter. For numeric parameters, such as those holding frequency or power values, the units are stated (e.g. W, Hz, dB, etc.). For <CPD> (character program data) parameters, the available choices are listed, separated by the "OR" symbol, " | ".

Angle brackets < ... > indicate that the enclosed parameter is described in more detail later in the text.

Example:

Parameters: <CPD>, <NRf>
input id, offset value

Valid values: input id: [A | B | C | RX]
offset value: real

The first line of the parameter definition states that the command takes two parameters. The first parameter is character program data and the second is a numeric value.

The semantic interpretation is given on the second line. The first parameter is the input to which an offset is to be applied; the second parameter is a number representing the offset value.

3. Valid values

Describes the type of number for numeric parameters, e.g. real or integer, and the allowable range, if applicable. If a parameter is defined as character program data, the allowable <CPD> strings are given. For string program data, the maximum number of characters is specified. In the above example, the first parameter can accept the strings A, B, C or RX; the second parameter accepts real numbers.

4. Description

Describes the purpose of the command. Where applicable, a cross-reference to the corresponding local (i.e. front panel) operation in Chapter 3 of the 6800 Operating Manual is provided. This cross reference is given in the form of a menu title, preceded by the title of the hard key which is used to access that menu; the page number can be easily found by referring to the Chapter 3 contents list.

5. Example.

An example of the use of the command is provided. Examples always use the short form of the command.

6. Query response

Query responses follow the same format as parameter definitions. The first line shows the response in terms of its IEEE 488.2 functional elements, and below it is given the semantics of the response.

Example:

Response: <NR2>
frequency start value

Returned values: frequency start value: real

Status reporting

The instruments within a GPIB system contain a set of registers that reflect the current state of the instrument and whether a particular event has occurred. It is also sometimes necessary for an instrument to generate an alert if that condition exists or if that event has occurred.

The 6800's status registers contain information about the condition of the instrument and its measurements. Using these registers it is possible to find out whether an error has occurred with a command, if averaging has completed for a particular measurement, if a measurement is out of limits, and other problems or conditions that may make a measurement unreliable. These registers can be used either by reading the contents directly when needed, or by configuring them to generate an interrupt signal (SRQ, service request) when the condition(s) of interest occurs. The status system consists of four readable registers:

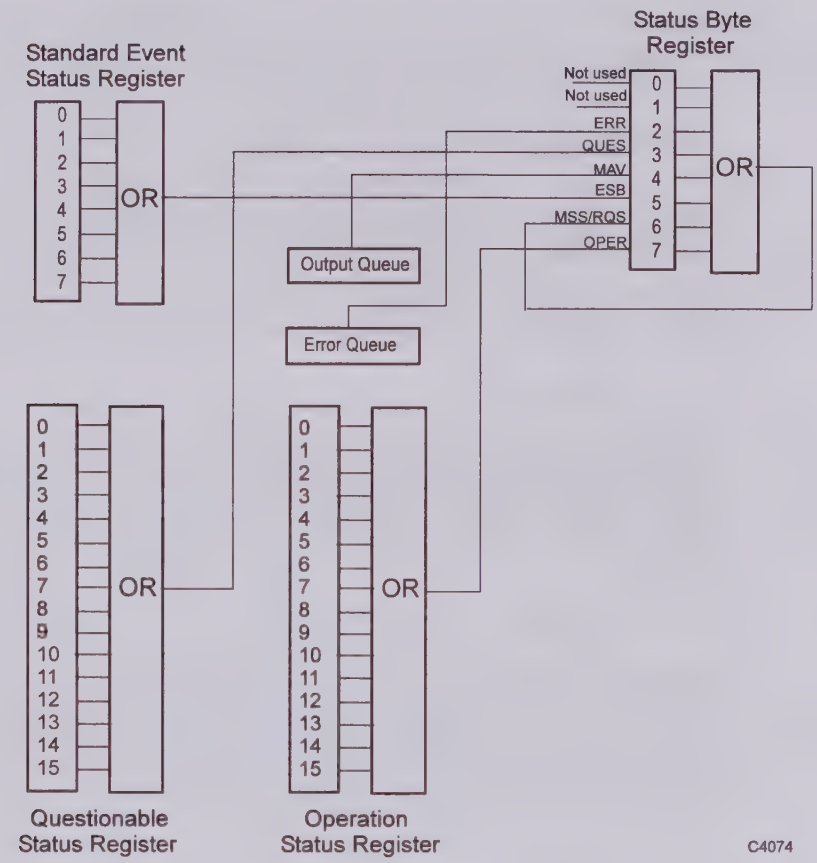


Fig. 1-1 Simplified status register structure

Status Byte Register. This is an 8-bit register that is used to represent particular conditions or events in an instrument. The Status Byte (defined by IEEE 488.1) is read by using the *STB? command or by serial poll. When read by serial poll, an SRQ is generated which interrupts the controller (described later). Associated with the Status Byte Register is the Service Request Enable Register, which allows control over which bits of the status byte contribute towards the generation of the SRQ signal. When read by *STB?, bit 6 of the Status Byte is known as the *master summary status* function (MSS), and is the OR function of the other 7 bits of the register.

Standard Event Status Register. This 8-bit register extends the status reporting structure to cover various other events, defined by IEEE 488.2. The register is read by *ESR? The Standard Event Status Enable Register allows control over which bits of the Standard Event Status Register affect the summary bit output (ESB). The summary bit is recorded in bit 5 of the instrument's Status Byte.

Operation Status Register. This is a 16-bit register, defined in SCPI, which further extends the status reporting structure by providing information about what the instrument is doing. It is read by the STATus:OPERation:CONDition? command. The summary bit output of the register is recorded in bit 7 of the Status Byte.

Questionable Status Register. This is another 16-bit register, also defined in SCPI, which gives information about factors affecting the quality of measurements or signal generation. It is read by the STATus:QUEStionable:CONDition? command. The summary bit output of the register is recorded in bit 3 of the Status Byte.

The *output queue* temporarily stores responses to query commands received by the instrument until they can be read by the controller. The *error queue* temporarily stores error messages. Each time the instrument detects an error, it places a message in the queue; each item contains an error number, defined in SCPI, and an error message. When the SYSTem:ERRor? query is sent, the message at the head of the error queue is moved to the output queue so it can be read by the controller.

The Operation and Questionable register structures consist of Condition, Event, Transition and Enable registers.

The condition registers continuously monitor the instrument's hardware and firmware status. Bits in a condition register are not latched but are updated in real time, so they represent the actual state of the instrument at all times, and are read by the above commands.

The bits of the event registers (read by STATus:OPERation:EVENt? and STATus:QUEStionable :EVENt?) are set on events. For example, the Averaging bit in the operation register only indicates if the measurement is being performed with averaging enabled, while the associated event register shows that the averaging has completed. A set of transition filters (Transition Register) control what type of change in a condition register will set the corresponding bit in the event register. The type of transition filter, negative, positive or both, is fixed for each bit. For example, the averaging bits in the Operation register structure have negative transition filters so that the bits in the event register are set when averaging is complete. When the event register bits are set they remain set, even if the corresponding condition bits change. They are reset after being read by the query commands STATus:OPERation:EVENt? and STATus:QUEStionable :EVENt?, or when the *CLS (clear status) common command is issued. Transition registers are read-write, and are unaffected by query commands or *CLS.

The ability of each bit in the event registers to affect the summary bit in the Status Byte Register can be enabled or disabled by corresponding bits in the event enable registers. These can be set and read by the commands/queries STATus:OPERation:ENABle\? and STATus:QUEStionable :ENABle\? The enabled bits are combined in a logical OR operation to produce the summary bit (summary bits are recorded in the instrument's status byte). Enable registers are cleared by *CLS.

The above status reading commands return the decimal number equivalent of the register contents.

The events and conditions that are monitored by the 6800's status registers, and the commands for reading and writing to them, are described in more detail in Appendix A.

As already stated, two techniques are used to interact with the status reporting structure:

Direct-read method. In many cases it is adequate and convenient for the controller simply to read the appropriate registers when necessary to determine the required status information. This technique does not involve the use of SRQ and therefore does not require any interrupt handling code in the application program. The following steps are used to monitor a condition:

1. Determine which register contains the bit that monitors the condition.
2. Send the query command that reads the register.
3. Examine the bit to see if the condition has changed.

The direct-read method works well when it is not necessary to know about changes the moment they occur. A program that uses this method to detect changes in a condition as soon as possible would need to continuously read the registers at very short intervals; the SRQ method is better suited for this type of need.

Service request (SRQ) method. In the SRQ method the instrument plays a more active role, in that it tells the controller when there has been a condition change without the controller asking. The following steps are required to monitor a condition:

1. Determine which register set and which of its bits monitors the condition.

2 Determine how that bit reports to the request service (RQS) bit of the Status Byte (some report directly while others may report indirectly through other register sets).

3. Send remote commands to enable the bit that monitors the condition and to enable the summary bits that report the condition to the RQS bit.

4. Enable the controller to respond to service requests.

When the condition changes, the instrument sets its RQS bit (bit 6) and the GPIB's SRQ line; the controller is informed of the change as soon as it occurs. Setting the SRQ line informs the controller that some device on the bus requires service. The GPIB program then instructs the controller to perform a serial poll; each device on the bus returns the contents of its Statue Byte

register in response to this poll. The device whose RQS bit is set to 1 is the device that requested service. After the Status Byte is read the RQS bit is reset to 0; the other bits are not affected.

Another reason for using SRQ is the need to detect errors in the various devices within the instrument. Since the timing of errors may not be known in advance, and it is not practical for the program to check the status of every device frequently, an interrupt handling routine can be used to detect and investigate any SRQ generated.

Chapter 2
GETTING STARTED

Contents

List of figures 2-1
Introduction 2-1
The remote operation command set 2-1
Preparing the 6800 for GPIB operation 2-3
Preparing the 6800 for RS-232 operation 2-3
Example: Low pass filter characterisation 2-4

List of figures

Fig. 2-1 Simultaneous measurement of insertion and return loss of a low pass filter 2-5
Fig. 2-2 Broadband insertion and return loss and narrowband insertion loss 2-8

Introduction

This chapter provides an introduction to GPIB programming of the 6800, including a worked example of a typical measurement. GPIB programmers may find it a worthwhile exercise to work through the example measurements given in the Getting Started Manual in order to gain familiarity with the front panel operation of the instrument.

The remote operation command set

The first point to notice when controlling the 6800 remotely is that there is no straightforward mapping between manual front panel operations and their remote command equivalents. The 6800's man-machine interface restricts the user's view of the instrument's functions by taking account of the active channel mode and active measurement, and making available appropriate function sub-sets as demanded by the context.

There is no corresponding mechanism for providing "context sensitive" remote commands. Instead, the commands are organised into "subsystems" of related functions, and the number of unique mnemonics required kept to a minimum by the use of compound headers. (See Chapter 1 for an introduction to compound headers and other command set conventions.)

To help programmers know where to look for a particular command, there follows a brief overview of the command subsystems.

Common commands

A selection of IEEE 488.2 common commands is provided. These all start with a " * " character. The most important of these is *RST, which places the instrument in a defined state. It is good practice to send *RST at the start of any GPIB program.

CHANnel

The CHANnel subsystem is used to:

Specify whether 1 or 2 channels are to be displayed
Specify which channel is to be the active one
Specify the channel type (i.e. scalar, fault location or spectrum analyzer)
Turn channel coupling on or off

DISPlay

The DISPlay subsystem is used to:

Control the backlight brightness and display contrast
Select the palette for colour displays
Define measurement and screen titles
Select start/stop or centre/span mode
Define source of x-axis (domain) annotation
Define frequency offset/scaling between the spectrum analyzer receiver and the display
Adjust vertical positioning and scaling for the measurement traces

SCALar, FLOCation and SANalyzer

Each channel type has an associated subsystem providing commands for setting up and acquiring measurements (including calibration). It is important to ensure that the active channel mode (CHANnel:MODE\?) is set appropriately before issuing commands from these subsystems.

MEASurement

The MEASurement subsystem is used to:

Specify the number of measurements to be displayed in the active channel
Specify which measurement in the active channel is to be the active one
Define the measurement (e.g. absolute power, power ratio, memory)
Select measurement format (units of vertical axis of graticule)
Display stored measurement traces (internal memory or floppy disk)
Save a measurement trace to internal memory or floppy disk

INPut

The INPut subsystem contains commands associated with the signal inputs, e.g. detection mode, detector correction, temperature correction, detector zeroing input offsets.

SOURce

The SOURce subsystrem contains all commands concerned with the control of the synthesized source, e.g. source mode, sweep range, sweep time, number of measurement points, levelling mode and RF on/off. Commands are also provided for user calibration of the source.

MARKer

The MARKer subsystem provides marker functions for scalar, fault location and spectrum analyzer measurements.

MMEMory

The MMEMory subsystem is used to:

Provide listings of floppy disk contents
Copy, delete and rename files (internal memory and floppy disk)
Format disks and creating directories
Transfer various types of data to/from the instrument over the GPIB interface

HARDcopy

The HARDcopy subsystem controls all available hard copy functions.

SYSTem

The SYSTem subsytem is used to:

Set up the instrument GPIB address
Install applications
Set the instrument date / time
Specify Country/Language/Keyboard settings
Specify the serial interface settings
Save / recall instrument settings
Provide service functions
Set the instrument to its default state (preset)

STEP

The STEP subsystem is used to set the amount that is added to or subtracted from a parameter's value when UP or DOWN is selected as a parameter instead of a numeric value. The step value can be set for various parameter type (e.g. frequency, power, time)/

STATus

Accesses the SCPI-compatible status reporting structure.

Preparing the 6800 for GPIB operation

Connection to the external controller is made via the rear panel GPIB connector. The 6800 GPIB system can operate in two modes, controller or talker/listener. In controller mode, the 6800 takes control of the bus in order to drive a plotter or an external source. This mode is selected by pressing the *[No External Controller]* soft key. Talker/listener mode must be selected before the instrument can accept commands from an external controller; in this case this is done by pressing the *[Controlled by GPIB]* soft key.

[UTILITY] ***[Remote Control]***	This menu allows the GPIB mode and instrument address to be set.
[Controlled by GPIB]	Enables the 6800 to be controlled by a GPIB controller.
[GPIB Address]	Specify a new GPIB address, if necessary.

Preparing the 6800 for RS-232 operation

Connection to the external controller is made via a 25-way D-type connector on the rear panel of the 6800 (pin assignments are given in Chapter-2 of the Operating Manual). For RS-232 operation, make the appropriate selection from the Remote menu. In addition, the interface parameters may need to be set to match the controller, and the type of data flow control specified.

[UTILITY] ***[Remote Control]*** ***[Controlled by RS232]***	Enables the 6800 to be controlled by an RS-232 controller.
[Set Up RS232]	This leads to the Set Up RS232 menu, where the Baud Rate, Parity and type of data flow control can be set, if required. Flow control can be set to either hardware control or software control. With the former method, dedicated control

lines (handshake signals) are used to signify ready or not ready for data. With software control, the control characters XON (start transmitting) and XOFF (stop transmitting) are used; control lines are ignored.

Note that the number of Stop Bits and Data Bits are defaulted when the instrument is being controlled via the RS-232, and cannot be set from this menu.

Example: Low pass filter characterisation

In this example, 6800 GPIB commands are stated without making any assumptions about the controller and programming language to be used. These commands, of course, will need to be incorporated into the program language statements of the target controller. Here are some examples of how this would be done in practice, using the reset command, *RST. The instrument address is assumed to be 8.

***RST**	Command as printed in the example.
PRINT @8:"*RST"	Controller using TBASIC^{R} programming language (TransEra Corporation).
OUTPUT 708;"*RST"	Controller using HTBASIC^{TM} programming language (TransEra Corporation).

The example will measure the broadband insertion and return loss responses of a 13 GHz low pass filter and simultaneously display the results on channel 1. Channel 2 will be used to display the narrowband insertion loss of the filter in order to determine the passband ripple. The set-up for this example is shown in Fig. 2-1. Note that either a 6823, 6824, 6843 or 6844 is needed in order to cover the required frequency range.

As far as GPIB control is concerned, a similar sequence of logical steps can be followed as described in the Getting Started Manual for the 6800.

It may sometimes be necessary to send a DEVICE CLEAR command, if the GPIB system fails to respond to *RST or appears to lock up. Examples of this command are as follows:

DEVICE CLEAR	Command as printed in the example.
WRITE GPIB CMD_SDC(8)	Controller using TBASIC$^{R.}$
CLEAR 708	Controller using HTBASIC^{TM}

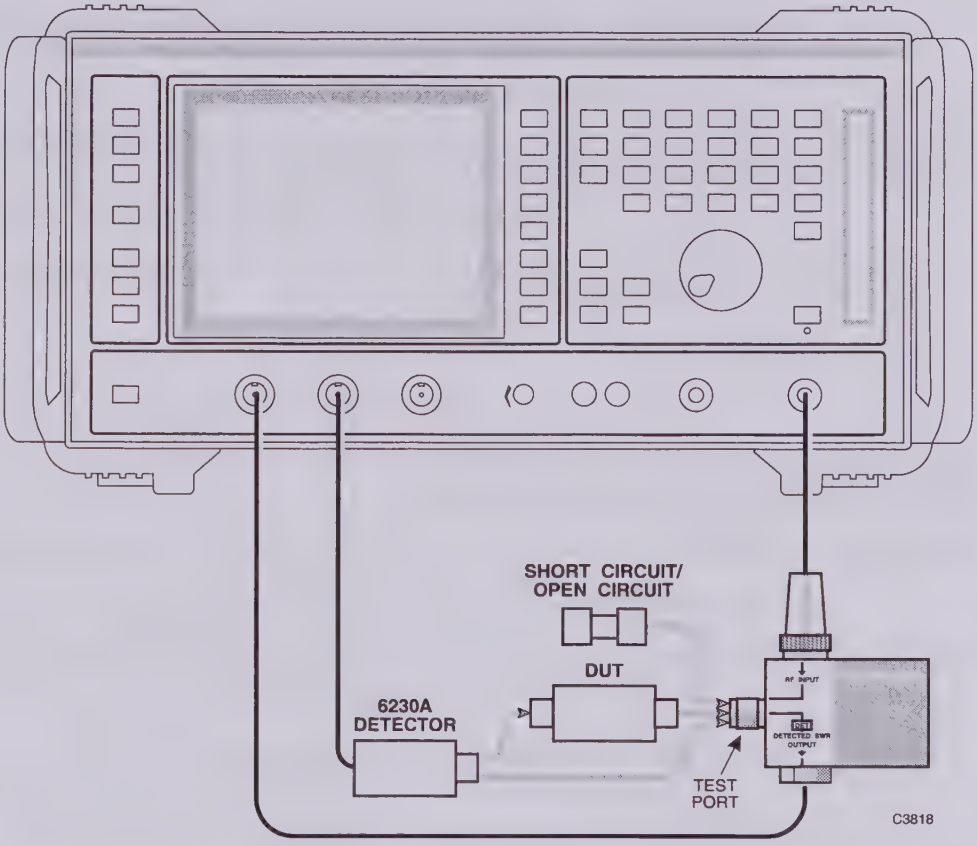


Fig. 2-1 Simultaneous measurement of insertion and return loss of a low pass filter

Step 1 Preset the instrument to a known state

DEVICE CLEAR

***RST** Preset the instrument:

Step 2 - Define the display configuration

Simultaneous broadband insertion and return loss measurements will be displayed on channel 1. Channel 2 will display the passband ripple (insertion loss) of the filter.

:CHANnel:NCHannels 2 Use the NCHannels (number of channels) function in the CHANnel sub-system to set up a dual channel display.

Remember that IEEE 488.2 requires a single space character between the command header and its parameter(s).

:CHANnel:ACTive 1 Make channel 1 the active channel.

Note that these two commands may be combined easily using the properties of compound headers described in Chapter 1. Thus:

CHAN:NCH 2;ACT 1

From now on, only the abbreviated forms will be given, and contractions that take advantage of the compound header structure will be used whenever possible.

Also note that for 6840 Series instruments the default mode for channel 1 is Spectrum Analyzer; to set it to Scalar use:

CHAN:MODE SCAL

:MEAS:NME 2	Set the number of measurements on the active channel (channel 1) to 2.

Step 3 - Define the measurements

:MEAS:ACT 1	Make measurement 1 on channel 1 the active measurement.
:MEAS:MEAS:POW B	Define measurement 1 to display power from input B. Return loss is measured by input A, and displayed as measurement 2; since input A is the default, no change is required here.
:CHAN:ACT 2;:MEAS:ACT 1	Make measurement 1 on channel 2 the active measurement.
:MEAS:MEAS:POW B	Set it up to display power from input B.

Step 4 - Define the source conditions

:CHAN:COUP OFF; ACT 1	Disable channel coupling, (so that a different source set-up may be defined for each channel), and make channel 1 the active channel.
:SOUR:FREQ:STOP 20E9	For the broadband measurements on channel 1, set the source stop frequency to 20 GHz. The start frequency defaults to 10 MHz for 6823/6824/6843/6844.
:CHAN:ACT 2 **:SOUR:FREQ:STOP 13E9**	Set the source stop frequency to 13 GHz.
:SOUR:RF ON	Switch on the RF.

Step 5 - Calibrate the measurement system

Calibrating the broadband insertion loss path

:INP:ZERO:AUTO ON	By default, detector autozeroing is disabled; this command turns it on.

Although detector autozero is enabled, it is necessary to perform a "full" zero when the detectors are changed. In a GPIB system it is good practice to perform a zero before a path calibration.

:INP:ZERO	Initiate a detector zero.

:CHAN:ACT 1;:MEAS:ACT 1 Make channel 1, measurement 1 active.

The insertion loss path is about to be calibrated. At this point in the program, the GPIB controller should issue a prompt to the user to make the through connection, then wait for confirmation from the user that the connection has been made.

:SCAL:PCAL:THR "PCL1" Perform the THRough path calibration, saving the calibration data in path cal store "PCL1".

Calibrating the broadband return loss path

:MEAS:ACT 2 Make measurement 2 on channel 1 the active measurement.

The program should prompt the user to connect a short circuit to the test port of the autotester, then wait for confirmation that the connection has been made.

:SCAL:PCAL:SHOR "PCL2" Perform the SHORt path calibration, saving the calibration data in path cal store "PCL2".

Again the program should prompt the user - this time to replace the short circuit with an open circuit.

:SCAL:PCAL:OPEN:MERG "PCL2" Perform the OPEN path calibration; the open cal data is averaged with the previously stored short data in path cal store "PCL2".

Calibrating the narrowband insertion loss path

:CHAN:ACT 2;:MEAS:ACT 1
:SCAL:PCAL:SEL "CAL_1;STAT ON Apply the through path cal to the narrow band insertion loss measurement displayed on channel 2.

Step 6 - Select appropriate scaling and format

:CHAN:ACT 1;:MEAS:ACT 1
:DISP:SCAL:RLEV 20 Sets the reference level for the broadband insertion loss measurement to +20 dB.

:MEAS:ACT 2
:MEAS:FORM VSWR
:DISP:SCAL:DIV 0.1 Display the return loss measurement in VSWR format, and set the scaling to 0.1/div (i.e. VSWR range of 1 to 2).

:CHAN:ACT 2;:MEAS:ACT 1
:DISP:SCAL:DIV 0.2 Selects more appropriate values of scaling for the narrowband insertion loss measurement displayed on channel 2 (see Fig. 2-2).

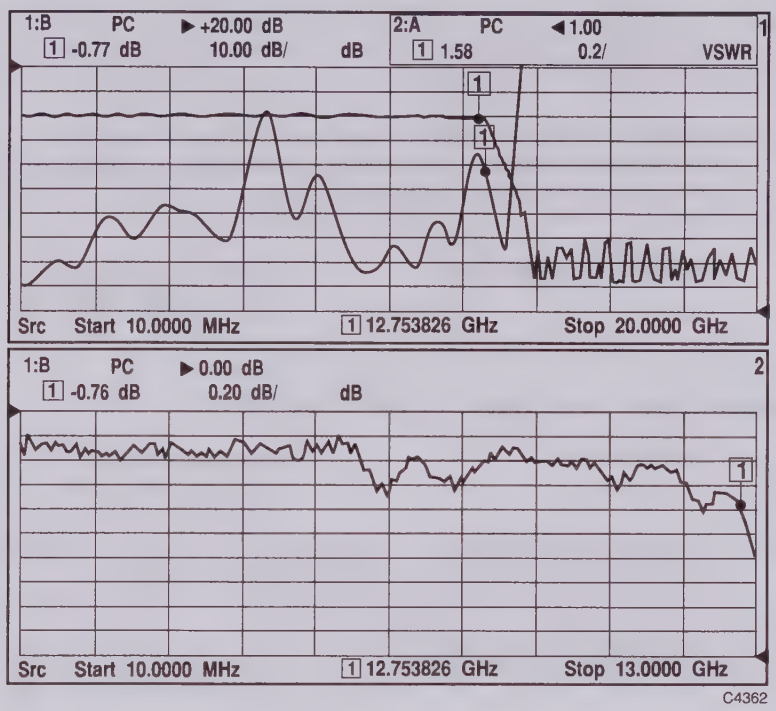


Fig. 2-2 Broadband insertion and return loss and narrowband insertion loss

Step 7 - Use markers to get detailed information about the measurement

The marker functions will now be used to determine the –3 dB cutoff point of the filter.

:CHAN:ACT 1;:MEAS:ACT 1

Ensure that the broadband insertion loss measurement is the active measurement.

:MARK:MAX;DELT ON

Place the active marker at the trace maximum, then switch on the delta marker. The delta marker will be placed at the active marker position.

:MARK:SEAR:DIR RIGH; TARG -3.0;RES?

This command causes the 6800 to search right from the current active marker position until the response value has dropped by 3 dB, relative to the delta marker response. A Boolean value is returned indicating whether the search has succeeded or failed.

:MARK:ACT:POS?

Assuming the search succeeded, this command returns the frequency at which the filter response has fallen by 3 dB.

Step 8 - Create a permanent record of the measurement results

Newcomers to the 6800 GPIB facilities will probably find it easier to create hard copy using a printer connected to the parallel interface, rather than a GPIB plotter. (The operations required to pass control to the 6800 temporarily so that it can drive a plotter are described under the :HARDcopy:PLOT command.)

:HARD:DEV "HP COLOUR DESKJET"	Select the HP Colour DeskJet printer driver.
:HARD:PORT PAR, ""	Select the Parallel port for hard copy output. Note that this command is unnecessary in this case since this port is the default for printers.
:HARD; *OPC?	Start the print, and wait for operation complete. *OPC? returns the value 1 when the last byte has been output to the printer.

Step 9 - Save the instrument settings for future use

Having just designed a program to set up the instrument from scratch, there seems little point in saving the set-up to a settings store. However, there are some aspects of the GPIB control of settings stores which are worth mentioning. It also provides an opportunity to utilise the MMEMory subsystem.

:MMEM:MSIS "C" **:SYST:SETT:SAVE "IFR_1"**	Save instrument settings to store IFR_1 in internal memory.

If it is necessary to transfer the instrument settings to the controller, this can be done by first saving to a specified store, as above, then transferring that store to the controller.

:MMEM:READ:SETT? "IFR_1"	Transfer the contents of settings store IFR_1 to the controller as binary data.

The final command in the example transfers the contents of a settings store to the GPIB controller as <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>. There is little to be gained by trying to modify the contents of the store, but the data may be saved for later transmission back to the instrument.

Chapter 3
6800 COMMAND SET

CONTENTS

COMMON COMMANDS SUBSYSTEM 3-3

CHANnel subsystem 3-11

DISPlay subsystem 3-15

FLOCation subsystem 3-37

HARDcopy subsystem 3-59

INPut subsystem 3-83

MARKer subsystem 3-91

MEASurement subsystem 3-123

MMEMory subsystem 3-147

SANalyzer subsystem 3-171

SCALar subsystem 3-203

SOURce subsystem 3-227

STATus subsystem 3-257

STEP subsystem 3-267

SYSTem subsystem 3-277

Common commands subsystem

*CLS
*ESE\?
*ESR?
*IDN?
*OPC\?
*OPT?
*PCB
*RST
*SRE\?
*STB?
*TRG
*TST?
*WAI

*CLS

Parameters: none

Description: Clear Status clears the Standard Event Status Register, the Error Queue, the Operation Status Event Register and the Questionable Status Event Register.

Example: `*CLS`

Clear the status reporting structure.

*ESE

Parameters: <NRf>

mask

Valid values: mask: integer. Valid values are 0 to 255. Values outside range are rejected and an error generated.

Description: The Standard Event Status Enable command sets the Standard Event Status Enable Register. This is an eight bit register.

See Appendix A for details.

Example: `*ESE 2`

Set the Standard Event Status Enable Register to 2 (00000010 in binary). This will allow RQC (Request Control) messages generated by the instrument to be reported in the Event Summary Bit. (RQC is issued by the instrument when it needs to take control of the bus to drive the plotter).

*ESE?

Parameters: none

Response: <NR1>

mask

Returned values: mask: integer. Values are in the range 0 to 255.

Description: Read the Standard Event Status Enable Register. This is an eight bit register.

See Appendix A for details.

Example: `*ESE?`

*ESR?

Parameters: none

Response: <NR1>

register contents

Returned values: register contents: integer. Values are in the range 0 to 255.

Description: Read the value of the Standard Event Status Register. This is an eight bit register.

See Appendix A for details.

Example: `*ESR?`

*IDN?

Parameters: none

Response: <ARBITRARY ASCII RESPONSE DATA>

Instrument Identification

Returned values: Instrument Identification: string

Description: The Identification Query command allows information about the instrument to be read.

The Instrument Identification is split into four fields:

- Manufacturer
- Model
- Serial number
- Software Part No. and Issue No.

Manufacturer returns 'IFR' unless altered to user selected text

Model returns the instrument model number in the form 68xx where:

68xx	Description	Frequency range
6821	Scalar Analyzer	3.0 GHz
6822	Scalar Analyzer	8.4 GHz
6823	Scalar Analyzer	20.0 GHz
6824	Scalar Analyzer	24.0 GHz
6825	Scalar Analyzer	46.0 GHz
6841	Microwave System Analyzer	4.2 / 3.0 GHz
6842	Microwave System Analyzer	20.0 / 8.4 GHz
6843	Microwave System Analyzer	20.0 / 20.0 GHz
6844	Microwave System Analyzer	24.0 / 24.0 GHz
6845	Microwave System Analyzer	46.0 / 46.0 GHz
6846	Spectrum/Scalar Analyzer	24.0 / 8.4 GHz
6847	Spectrum/Scalar Analyzer	26.5 / 20.0 GHz
6848	Spectrum/Scalar Analyzer	20.0 / 3.0 GHz

Serial number is in the form ssssss/sss where s is an ASCII digit in the range 0 to 9.

Software Part No. and Issue No. is in the form ppppp/ppp/ii.ii where p and i are ASCII digits in the range 0 to 9.

Example: `*IDN?`

Read information on the instrument.

Example response: IFR,6844,123456/123,44540/026/01.00

*OPC

Parameters: none

Description: The Operation Complete command sets the Operation Complete bit in the Standard Event Status Register when execution of all overlapped commands have completed.

This command is really only useful after an overlapped command when it will indicate when that command has been completed. Other (non-overlapped) commands can be executed whilst the overlapped command is still being executed. If there is more than one overlapped command being executed, the Operation Complete bit will only be set once all of the overlapped commands complete.

*OPC should be the final <PROGRAM MESSAGE UNIT> of the <PROGRAM MESSAGE>.

See Appendix D for a list of overlapped commands.

Example:
```
:HARD:PLOT; *OPC
```

Initiate a plot of the currently displayed measurements. The Operation Complete bit will be set in the Standard Event Status Register when the instrument has finished printing.

*OPC?

Parameters: none

Response: <NR1>

operation complete

Returned values: operation complete: integer. Value is 1

Description: The Operation Complete Query returns a '1' when all overlapped commands have completed.

This command is really only useful after an overlapped command when it will indicate when that command has been completed.

*OPC? should be the final <QUERY MESSAGE UNIT> of the <PROGRAM MESSAGE>.

See Appendix D for a list of overlapped commands.

Example:
```
:HARD:BUIL; *OPC?
```

Initiate a print of the instrument's build state. When the instrument has finished printing, the value '1' will be placed in the output queue.

*OPT?

Parameters: none

Response: <ARBITRARY ASCII RESPONSE DATA>

options

Returned values: options: string

Description: Read hardware options present. If no options are present a single "0" is returned otherwise the response is a number of strings separated by commas. If the option is present it will return the string shown below, if it is not present nothing is returned for that option.

String	Meaning
MON	Monochrome display (colour is standard)
70DB	70 dB Attenuator (20 GHz)
90DB	90 dB Attenuator (24 GHz)
110DB	110 dB Attenuator (3 GHz)
132DB	132 dB Attenuator (3 GHz)
70DB46G	70 dB Attenuator (46 GHz)
WBDM	AM & Wideband FM Demodulator
EMIX	External Mixer
FCON	Field replaceable connectors
GDLY	Group Delay
FM	Source Frequency Modulation
EFM	Source FM (external modulation only)
HSO	High Power Source

Note that :SYST:OPT? can be used instead if a bit-field is easier to work with than this string format.

The options FM and EFM are mutually exclusive. If the option is present to generate internal FM then 'FM' will be reported otherwise 'EFM' will be reported, indicating that FM is available using an external modulating input.

Example: `*OPT?`

Example response: 90DB,EMIX

*PCB

Parameters: <NRf>[,<NRf>]

controller address[,secondary address]

Valid values: controller address: integer. Valid values are 0 to 30. Values outside range are rejected and an error generated.

secondary address: integer. Valid values are 0 to 30. Values outside range are rejected and an error generated.

Description: Any hard copy output, when being controlled over GPIB and the hard copy device is on the GPIB, requires the instrument to act temporarily as the GPIB controller. *PCB is used to tell the instrument the address of the System Controller so that control can be passed back to it on completion of the command. This command will generate an error if received over the serial interface.

Example: `*PCB 21`

Inform the instrument that the system controller's GPIB address is 21.

*RST

Parameters: none

Description: Reset the instrument. This command places the instrument in its default state. See Appendix C for details on which parameters are not affected by preset.

If the Remote control system fails to respond to *RST, it may be cleared by sending a DEVICE CLEAR command. It is good practice to precede *RST with DEVICE CLEAR.

Note that *RST is not identical to :SYST:PRES.

Example:
```
*RST
```
Reset instrument to known state.

*SRE

Parameters: <NRf>

mask

Valid values: mask: integer. Valid values are 0 to 255. Values outside range are rejected and an error generated.

Description: Set the Service Request Enable Register. This is an eight bit register.

See Appendix A for details.

Example:
```
*SRE 32
```
Set the Service Request Enable Register to 32 (0010 0000 in binary) to enable service requests when the Standard Event Status Register Summary Bit is set.

*SRE?

Parameters: none

Response: <NR1>

mask

Returned values: mask: integer. Values are in the range 0 to 255.

Description: Read the Service Request Enable Register. This is an eight bit register.

See Appendix A for details.

Example:
```
*SRE?
```

*STB?

Parameters: none

Response: <NR1>

status byte

Returned values: status byte: integer. Values are in the range 0 to 255.

Description: Read the Status Byte Register. This is an eight bit register. Bit 6 of the register contains the Master Summary Status.

See Appendix A for details.

Example:
```
*STB?
```

*TRG

Parameters: none

Description: Trigger command. This command is equivalent to Group Execute Trigger.

Example:
```
:SYST:TRIG MEAS; *TRG
```

Place instrument into Trigger Measurement mode and trigger the measurement.

*TST?

Parameters: none

Response: <NR1>

self test completed

Returned values: self test completed: integer. Values are in the range 0 to 1.

Description: Self Test Query. This performs a self test (identical to the power on test) and returns a value of 0 if the test passes and 1 if it fails.

If the test fails, use the :SYST:TEST:... commands to read the test results.

Example:
```
*TST?
```

Perform a self test.

*WAI

Parameters: none

Description: The Wait to Continue command inhibits execution of a command until the execution of all overlapped commands has been completed.

This command is really only useful after an overlapped command when it will hold off further commands until that command has been completed. If there is more than one overlapped command being executed, the next command will be held off until all of the overlapped commands complete.

See Appendix D for a list of overlapped commands.

Example:
```
:CHAN:ACT 1;:HARD:PLOT; :CHAN:ACT 2; *WAI;:HARD:PLOT
```

*Initiate a plot of the current measurement and inhibit further commands until the plot has finished. In the example above, without the *WAI an error would occur when the second plot was started because the first one would still be in progress and only one plot can be performed at a time.*

CHANnel subsystem

CHANnel
- **ACTive\?**
- **COUPling\?**
- **MODE\?**
- **NCHannels\?**

:CHANnel :ACTive

Parameters: <NRf>

active channel number

Valid values: active channel number: integer. Valid values are 1 to 2. Values outside range are rejected and an error generated.

Description: Set the active channel. Either channel 1 or channel 2 may be made active. The active channel is always displayed. If only one channel is currently being displayed and the other channel is made active then the currently active channel will be replaced by the other channel.

Example: `:CHAN:ACT 2`

Make channel 2 active.

:CHANnel :ACTive?

Parameters: none

Response: <NR1>

active channel number

Returned values: active channel number: integer. Values are in the range 1 to 2.

Description: Determine which channel is the active channel.

Example: `:CHAN:ACT?`

:CHANnel :COUPling

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set channel coupling on or off. If channels are coupled then channel wide parameters, such as start and stop domains, set on the active channel are mirrored on the inactive channel. With channel coupling off it is possible to set up the two channels independently.

Only two scalar channels can be coupled. If any channel is not scalar then this command will generate an error. Channel coupling will be turned off automatically if either channel is changed to be a non-scalar channel.

Example: `:CHAN:COUP OFF`

Disable channel coupling.

:CHANnel :COUPling?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if channel coupling is on or off.

Example:
```
:CHAN:COUP?
```

:CHANnel :MODE

Parameters: <CPD>

channel mode

Valid values: channel mode: [SCALar | FLOCation | SANalyzer]. Values other than those stated are rejected and an error generated.

Description: Set the mode of the currently active channel. Each channel can be set to perform scalar, fault location, or spectrum analyzer measurements.

An error will be generated if the mode is not supported by the instrument.

Example:
```
:CHAN:MODE FLOC
```

Make the active channel a fault location channel.

:CHANnel :MODE?

Parameters: none

Response: <CRD>

channel mode

Returned values: channel mode: [SCAL | FLOC | SAN]

Description: Determine the mode of the active channel.

Example:
```
:CHAN:MODE?
```

:CHANnel :NCHannels

Parameters: <NRf>

number of channels

Valid values: number of channels: integer. Valid values are 1 to 2. Values outside range are rejected and an error generated.

Description: Set the number of channels displayed. If one channel is displayed, it will always be the active channel.

Example: `:CHAN:NCH 2`

Display two channels.

:CHANnel :NCHannels?

Parameters: none

Response: <NR1>

number of channels

Returned values: number of channels: integer. Values are in the range 1 to 2.

Description: Determine the number of channels displayed.

Example: `:CHAN:NCH?`

DISPlay subsystem

DISPlay
- BLIGht\?
- CENTer\?
- CENTre\?
- CONTrast\?
- CPALette\?
- CSPan\?
- GRATicule
  - [MODE]\?
  - OFFSet\?
  - ORDer\?
  - SCALing\?
- MTITle
  - [STATe]\?
  - STRing\?
- SCALing
  - AUTO\?
  - DIVision\?
  - POSition\?
  - RLEVel\?
- SPAN\?
- STARt\?
- STITle
  - [STATe]\?
  - STRing\?
- STOP\?

:DISPlay :BLIGht

Parameters: <CPD>

backlight brightness

Valid values: backlight brightness: [OFF | MINimum | LOW | HIGH | MAXimum]. Values other than those stated are rejected and an error generated.

Description: Set the backlight brightness.

When using the instrument outside, the backlight on the monochrome LCD is not required and can be turned off by using :DISP:BLIG OFF. When using inside, the backlight should be turned on and set as appropriate to the ambient lighting conditions.

The backlight cannot be turned off on an instrument with a colour LCD. Selecting OFF will give an error.

Example: `:DISP:BLIG MAX`

Set the display backlight to Full Brightness.

:DISPlay :BLIGht?

Parameters: none

Response: <CRD>

backlight brightness

Returned values: backlight brightness: [OFF | MIN | LOW | HIGH | MAX]

Description: Read the backlight brightness.

Example: `:DISP:BLIG?`

:DISPlay :CENTer

Parameters: <NUMERIC VALUE>

centre value

Valid values: centre value: real

frequency (Hz), power (dBm), or distance (m or ft)

Suffix: centre value: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance	mm, m, km	m
Distance	ft	ft

Description: Set the display centre domain value.

For a scalar channel, the value will be interpreted as a frequency value (Hz) or a power value (dBm) depending on the source mode of the channel. This command sets the user display centre value. To use this value, select user graticule using :DISP:GRAT USER. The source is not affected by setting the user display centre value.

For a fault location channel the value will be distance in metres or feet. This command is only valid in a fault location channel when zoomed mode is enabled. Values entered without suffixes will be treated as m or ft dependant on the setting of :FLOC:UNIT.

For a spectrum analyzer channel, the value will be frequency (Hz). This command cannot be used if the channel is in a demodulation mode; an error will be generated. This command sets the user display centre value. To use this value, select user graticule using :DISP:GRAT USER. The source, tracking generator, or receiver frequencies are not affected by setting the user display centre value.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:CENT 12.4E9`

Set centre frequency to 12.4 GHz.

:DISPlay :CENTer?

Parameters: none

Response: <NR2>

centre value

Returned values: centre value: real

Description: Read the display centre domain value.

Example: `:DISP:CENT?`

:DISPlay :CENTre

Parameters: <NUMERIC VALUE>

centre value

Valid values: centre value: real

frequency (Hz), power (dBm), or distance (m or ft)

Suffix: centre value: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft

Description: Set the display centre domain value.

For a scalar channel, the value will be interpreted as a frequency value (Hz) or a power value (dBm) depending on the source mode of the channel. This command sets the user display centre value. To use this value, select user graticule using :DISP:GRAT USER. The source is not affected by setting the user display centre value.

For a fault location channel the value will be distance in metres or feet. This command is only valid in a fault location channel when zoomed mode is enabled. Values entered without suffixes will be treated as m or ft dependant on the setting of :FLOC:UNIT.

For a spectrum analyzer channel, the value will be frequency (Hz). This command cannot be used if the channel is in a demodulation mode; an error will be generated. This command sets the user display centre value. To use this value, select user graticule using :DISP:GRAT USER. The source, tracking generator, or receiver frequencies are not affected by setting the user display centre value.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:CENT 12.4E9`

Set centre frequency to 12.4 GHz.

:DISPlay :CENTre?

Parameters: none

Response: <NR2>

centre value

Returned values: centre value: real

Description: Read the display centre domain value.

Example: `:DISP:CENT?`

:DISPlay :CONTrast

Parameters: <NUMERIC VALUE>

contrast

Valid values: contrast: integer. Valid values are 0 to 100. Values outside range are clipped.

Suffix: contrast: A suffix of pct is accepted for a value of percent. If no suffix is entered then the default suffix of pct is assumed.

Description: Set the contrast between 0% and 100%. This will affect the viewing angle of the display.

This command will not be accepted on instruments with a colour display.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:CONT 50`

Set display contrast to 50%.

:DISPlay :CONTrast?

Parameters: none

Response: <NR1>

contrast

Returned values: contrast: integer. Values are in the range 1 to 100.

Description: Read the display contrast.

This command will not be accepted on instruments with a colour display.

Example: `:DISP:CONT?`

:DISPlay :CPALette

Parameters: <CPD>

palette

Valid values: palette : [COLour | WHITe | BLACk | GREen | MONochrome]. Values other than those stated are rejected and an error generated.

Description: Set the colour palette on an instrument equipped with the colour LCD. This command will generate an error if it is received on an instrument fitted with the monochrome LCD. Use *OPT? or :SYST:OPT? to determine which LCD is fitted.

COLour gives a normal colour display
WHITe gives white on black display
BLACk gives black on white display
GREen gives green on black display
MONochrome gives a greyscale display

Example:
```
:DISP:CPAL GRE
```
Select green on black display

:DISPlay :CPALette?

Parameters: none

Response: <CRD>

palette

Returned values: palette : [COL | WHIT | BLAC | GRE | MON]

Description: Determine the colour palette in use on an instrument equipped with the colour LCD. This command will generate an error if it is received on an instrument fitted with the monochrome LCD. Use *OPT? or :SYST:OPT? to determine which LCD is fitted.

Example:
```
:DISP:CPAL?
```

:DISPlay :CSPan

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: For a scalar channel, this selects centre / span display (ON) or start / stop display (OFF). This applies to both frequency and power domains. An error will be given if anything other than user graticule has been selected using :DISP:GRAT USER.

For a fault location channel, this selects centre / span display (ON) or start / stop display (OFF). This command is only valid in a fault location channel when zoomed mode is enabled.

For a spectrum analyzer channel, this selects centre / span display (ON) or start / stop display (OFF). If the channel is set up to a demodulation mode, this command will have no effect and an error will be given since in a demodulation mode the display is always start-stop time.

Example: `:DISP:CSP ON`

Set display to centre / span.

:DISPlay :CSPan?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the display of the channel is in start / stop mode or centre / span mode. This will always return the current state even if the instrument is in a state where the user cannot set the state.

Example: `:DISP:CSP?`

Determine if in centre / span or start / stop display mode.

:DISPlay :GRATicule [:MODE]

Parameters: <CPD>

graticule source

Valid values: graticule source : [SOURce | RX | OSOurce | USER]. Values other than those stated are rejected and an error generated.

Description: For a spectrum analyzer channel this selects where the domain annotation is derived from.

Selecting SOURce gives the tracking generator frequency range.

Selecting RX gives the receiver range.

Selecting USER allows the user to display any user entered range. If the channel is in a demodulation mode, this parameter is not used since the display will have an x-axis of time with a start of zero and a stop depending on the timebase value.

Selecting OSO will give an error.

For a scalar channel this selects where the domain annotation is derived from. This can be set to display the source range (SOUR), the source range scaled/offset (OSO), or an arbitrary user entered range (USER). Selecting RX will give an error.

For a fault location channel, the command is ignored and an error generated.

Example: `:DISP:GRAT RX`

Set display domain annotation to the receiver start/stop frequencies.

:DISPlay :GRATicule [:MODE]?

Parameters: none

Response: <CRD>

graticule source

Returned values: graticule source : [SOUR | RX | OSO | USER]

Description: Determine where the displayed domain values are derived from. This command applies to spectrum analyzer and scalar channels only. For a fault location channel type the command is ignored and an error generated.

Example: `:DISP:GRAT?`

:DISPlay :GRATicule :OFFSet

Parameters: <NUMERIC VALUE>

offset

Valid values: offset: real

Suffix: offset: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm

Description: Set the frequency/power offset from the source to the display for the active channel. This command will generate an error if the active channel is not a scalar channel. Note that this value will only be used if Graticule Mode is OSO.

Example: `:DISP:GRAT:OFFS -5.0E+9`

Offset source frequency relative to displayed frequency by -5 GHz.

:DISPlay :GRATicule :OFFSet?

Parameters: none

Response: <NR2>

offset

Returned values: offset: real

Description: Determine the frequency/power offset from the source to the display for the active channel. The offset value is in Hz/dBm. This command will generate an error if the active channel is not a scalar channel.

Example: `:DISP:GRAT:OFFS?`

:DISPlay
:MTITle
:STRing

Parameters: <STRING PROGRAM DATA>

measurement title string

Valid values: measurement title string: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

Description: Send a measurement title for display on the active measurement. Use :DISP:MTIT ON to display the measurement title.

Example: `:DISP:MTIT:STR "13.6 GHz LPF"`

Set the measurement title to 13.6 GHz LPF.

:DISPlay
:MTITle
:STRing?

Parameters: none

Response: <STRING RESPONSE DATA>

measurement title string

Returned values: measurement title string: string. Maximum length of 20 characters excluding quotes.

Description: Read the measurement title for the active trace.

Example: `:DISP:MTIT:STR?`

:DISPlay :SCALing :AUTO

Parameters: <CPD>

autoscale mode

Valid values: autoscale mode : [ONCE | CONTinuous | OFF]. Values other than those stated are rejected and an error generated.

Description: Set the autoscale mode for the active measurement:

ONCE	Immediate one off autoscale
CONTinuous	Enable continuous autoscaling (scalar channel only)
OFF	Disable continuous autoscaling

Autoscaling calculates optimum settings for scaling and reference level to place the trace on the screen.

Continuous autoscaling is performed at the end of a measurement update, in other words, after all traces have been updated with live data. Continuous autoscaling is only available on scalar channels.

One off autoscaling is performed immediately on receipt of the command. It does not wait until the end of the measurement update. If continuous autoscaling was enabled, it is turned off.

This command is not valid for a spectrum analyzer channel, an error will be reported.

Example: `:DISP:SCAL:AUTO CONT`

Enable continuous autoscaling.

:DISPlay :SCALing :AUTO?

Parameters: none

Response: <CRD>

continuous autoscale status

Returned values: continuous autoscale status : [CONT | OFF]

Description: Determine if continuous autoscaling is on. The scaling mode ONCE will never be returned by the query form of the command.

This command is not valid for a spectrum analyzer channel, an error will be reported.
Whilst this command is valid for fault location channels, it will always return OFF since continuous autoscaling is only valid for scalar channels.

Example: `:DISP:SCAL:AUTO?`

:DISPlay :SCALing :DIVision

Parameters: <NUMERIC VALUE>

per division

Valid values: per division : real

Suffix: per division: The suffix that is accepted depends on the format of the trace:

Format	Allowable suffixes	Default suffix (if none entered)
Log power	dB	dB
dBµV	dBuV	dBuV
Lin power	pW, nW, uW, mW, W	W
Volts	nV, uV, mV, V	V
Frequency (FM demod)	Hz, kHz, MHz	Hz
Time (delay)	ps, ns, us, ms, s	s

Description: Set the graticule vertical scaling on the active measurement. If autoscaling is on (continuous), it will be turned off.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:SCAL:DIV 10`

Set the vertical scaling to 10 dB per division (assuming log format).

:DISPlay :SCALing :DIVision?

Parameters: none

Response: <NR3>

per division

Returned values: per division: real

Description: Determine the graticule vertical scaling for the active measurement.

Example: `:DISP:SCAL:DIV?`

:DISPlay :SCALing :POSition

Parameters: <NRf>

reference position

Valid values: reference position: integer. Valid values are 0 to 10. Values outside range are clipped.

Description: Set the reference position for the active measurement. . If autoscaling is on (continuous), it will be turned off. This command will give an error if the active channel is a spectrum analyzer channel since the reference position is fixed. The top line of the graticule is line 10, the bottom line is line 0.

Example: `:DISP:SCAL:POS 5`

Set the centre graticule line to be the reference line.

:DISPlay :SCALing :POSition?

Parameters: none

Response: <NR1>

reference position

Returned values: reference position: integer. Values are in the range 0 to 10.

Description: Determine the reference position for the active measurement.

Example: `:DISP:SCAL:POS?`

:DISPlay :SCALing :RLEVel

Parameters: <NUMERIC VALUE>

reference level

Valid values: reference level: real

Suffix: reference level: The suffix that is accepted depends on the format of the trace:

Format	Allowable suffixes	Default suffix (if none entered)
Log power (absolute)	dBm	dBm
Log power (relative)	dB	dB
dBµV	dBuV	dBuV
Lin power	pW, nW, uW, mW, W	W
Volts	nV, uV, mV, V	V
Percent	PCT	PCT
Time (delay)	ps, ns, us, ms, s	

Description: Set the reference level for the active measurement. If autoscaling is on (continuous), it will be turned off.
Reference level units depend on the display format.
For a spectrum analyzer channel setting the reference level may alter the input attenuation if coupling is enabled (see :SAN:ATT:AUTO).

The reference level cannot be set if the active measurement is a spectrum analyzer demodulated waveform. An error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:DISP:SCAL:RLEV -10
```
Set the reference level to -10 dB (assuming log format).

:DISPlay :SCALing :RLEVel?

Parameters: none

Response: <NR3>

reference level

Returned values: reference level: real

Description: Read the reference level for the active measurement.

Example:
```
:DISP:SCAL:RLEV?
```

:DISPlay :SPAN

Parameters: <NUMERIC VALUE>

span value

Valid values: span value: real

frequency (Hz), power (dB), or distance (m or ft)

Suffix: span value: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dB	dB
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft

Description: Set the display span value.

For a scalar channel, the value will be interpreted as a frequency value (Hz) or a power value (dB) depending on the source mode of the channel. This command sets the user display span. To use this value, select user graticule using :DISP:GRAT USER. The source is not affected by setting the user display span.

For a fault location channel the value will be distance in metres or feet. This command is only valid in a fault location channel when zoomed mode is enabled. Values entered without suffixes will be treated as m or ft dependant on the setting of :FLOC:UNIT.

For a spectrum analyzer channel, the value will be frequency (Hz). This command cannot be used if the channel is in a demodulation mode; an error will be generated. This command sets the user display span. To use this value, select user graticule using :DISP:GRAT USER. The source, tracking generator, or receiver frequencies are not affected by setting the user display span.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:SPAN 10E9`

Set the frequency span to 10 GHz (assuming the domain is frequency).

:DISPlay :SPAN?

Parameters: none

Response: <NR2>

span value

Returned values: span value: real

Description: Read the display span value.

Example: `:DISP:SPAN?`

:DISPlay :STARt

Parameters: <NUMERIC VALUE>

start domain

Valid values: start domain: real

frequency (Hz), power (dBm), or distance (m or ft)

Suffix: start domain: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft

Description: Set the display start domain value.

For a scalar channel, the value will be interpreted as a frequency value (Hz) or a power value (dBm) depending on the source mode of the channel. This command sets the user display start value. To use this value, select user graticule using :DISP:GRAT USER. The source is not affected by setting the user display start value.

For a fault location channel the value will be distance in metres or feet. This command is only valid in a fault location channel when zoomed mode is enabled. Values entered without suffixes will be treated as m or ft dependant on the setting of :FLOC:UNIT. If zoom mode is off, the display start distance is always zero.

For a spectrum analyzer channel, the value will be frequency (Hz). This command cannot be used if the channel is in a demodulation mode; an error will be generated. This command sets the user display start value. To use this value, select user graticule using :DISP:GRAT USER. The source, tracking generator, or receiver frequencies are not affected by setting the user display start value.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:STAR 1E9`

Set 1 GHz start frequency.

:DISPlay :STARt?

Parameters: none

Response: <NR2>

start domain

Returned values: start domain: real

Description: Read the display start domain value.

Example: `:DISP:STAR?`

:DISPlay
:STITle
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set screen title on or off.

Example: `:DISP:STIT ON`

Enable screen title display.

:DISPlay
:STITle
[:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Read whether screen title is displayed or not.

Example: `:DISP:STIT?`

:DISPlay
:STITle
:STRing

Parameters: <STRING PROGRAM DATA>

screen title string

Valid values: screen title string: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Send a screen title for display. Use :DISP:STIT ON to display the screen title.

Example: `:DISP:STIT:STR "This is a SCREEN TITLE"`

Set the screen title to This is a SCREEN TITLE.

:DISPlay
:STITle
:STRing?

Parameters: none

Response: <STRING RESPONSE DATA>

screen title string

Returned values: screen title string: string. Maximum length of 30 characters excluding quotes.

Description: Read the screen title.

Example: `:DISP:STIT:STR?`

:DISPlay :STOP

Parameters: <NUMERIC VALUE>

stop domain

Valid values: stop domain: real

frequency (Hz), power (dBm), or distance (m or ft)

Suffix: stop domain: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft

Description: Set the display stop domain value.

For a scalar channel, the value will be interpreted as a frequency value (Hz) or a power value (dBm) depending on the source mode of the channel. This command sets the user display stop value. To use this value, select user graticule using :DISP:GRAT USER. The source is not affected by setting the user display stop value.

For a fault location channel the value will be distance in metres or feet. This command is only valid in a fault location channel when zoomed mode is enabled. Values entered without suffixes will be treated as m or ft dependant on the setting of :FLOC:UNIT. If zoom mode is off, the display stop is set by either the range value or the frequency sweep of the source.

For a spectrum analyzer channel, the value will be frequency (Hz). This command cannot be used if the channel is in a demodulation mode; an error will be generated. This command sets the user display stop value. To use this value, select user graticule using :DISP:GRAT USER. The source, tracking generator, or receiver frequencies are not affected by setting the user display stop value.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:DISP:STOP 25E9`

Set 25 GHz stop frequency.

:DISPlay :STOP?

Parameters: none

Response: <NR2>

stop domain

Returned values: stop domain: real

Description: Read the display stop domain value.

Example: `:DISP:STOP?`

FLOCation subsystem

```
FLOCation
        ATTenuation\?
        CALibration
                RESTore
                [SAVE]
                SELect
                VERify?
        DBASe
                CABLes
                        [LIST]?
                        NUMBer?
                COPY
                        [DISK]
                ID
                        [DISK]?
                        STORe\?
                MANufacturer
                        [LIST]?
                        NUMBer?
                READ
                        [DISK]?
                        STORe?
                SELect
                        [DISK]
                        STORe
                [STATe]\?
                WRITe
                        STORe
        ENTRy\?
        MCORrection\?
        MEDium\?
        RANGe\?
        RVELocity\?
        UNITs\?
        WGCutoff\?
        WINDow\?
        ZOOM\?
```

:FLOCation :ATTenuation

Parameters: <NUMERIC VALUE>

attenuation

Valid values: attenuation: real

Suffix: attenuation: The suffix that is accepted depends on the FLOC:UNIT command - whether metres or feet have been selected for distances:

FLOC:UNIT	Allowable suffixes	Default suffix (if none entered)
metres	dB/m	dB/m
feet	dB/ft	dB/ft

Description: Set the attenuation per unit length of the transmission line under test for the active channel. The value is either dB/m or dB/ft, depending on the setting of FLOC:UNIT..

This command will generate an error if the fault location cable database is enabled.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:FLOC:ATT 0.1`

Set the attenuation per unit length to 0.1 dB/m.

:FLOCation :ATTenuation?

Parameters: none

Response: <NR2>

attenuation

Returned values: attenuation: real

Description: Determine the attenuation per unit length of the transmission line under test for the active channel. The value will always be in units of dB/m or dB/ft depending on the setting of FLOC:UNIT. This command will always return the current value regardless of whether it was set by the user or the fault location cable database.

Example: `:FLOC:ATT?`

:FLOCation
:CALibration
:RESTore

Parameters: none

Description: Using the calibration store that is currently associated with the active channel, the fault location setup is recalled so that the calibration is made valid. This will allow measurements to be performed.

An error will be generated if there is no valid calibration store.

Example: `:FLOC:CAL:REST`

Re-establish measurement conditions that were stored with the calibration.

:FLOCation
:CALibration
[:SAVE]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate a fault location calibration by saving the current measurement data and conditions to the specified fault location calibration store. That store is then associated with the active channel.

Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set. If averaging is off only one sweep will be performed.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Calibrations cannot be directly stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:FLOC:CAL "PCL1"`

Perform a fault location calibration and save to store CAL1 *in internal memory.*

:FLOCation
:CALibration
:SELect

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Assign a fault location calibration store to the active measurement. This may be used to allow both channels to use the same fault location calibration.
Fault location calibrations cannot be stored on the floppy disk. However, they may be copied to a floppy using MMEM:COPY.

An error will be given if the calibration store does not contain a fault location calibration.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Example: `:FLOC:CAL:SEL "PCL2"`

Assign cal store PCL2 *to the measurement.*

:FLOCation
:CALibration
:VERify?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <BOOLEAN RESPONSE DATA>

valid fault location cal

Description: Determine if the specified calibration store contains a fault location calibration. Since calibration stores are capable of holding either scalar path cals or fault location cals, this allows the user to confirm that a store actually contains fault location calibration data.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Example: `:FLOC:CAL:VER? "PCL4"`

:FLOCation
:DBASe
:CABLes
[:LIST]?

Parameters: <STRING PROGRAM DATA>, <CPD>, <STRING PROGRAM DATA>

manufacturer subset, medium subset, cable subset

Valid values: manufacturer subset: string. Maximum length of 32 characters excluding quotes. Excess characters will be ignored.

medium subset: [COAX | WAVeguide | ANY]. Values other than those stated are rejected and an error generated.

cable subset: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

Response: <ARBITRARY ASCII RESPONSE DATA>

cable data

Description: List cables on the fault location cable database. The cable database disk must be present.

Either all cables can be listed or the list can be reduced by specifying a manufacturer (or part of, e.g. "A*" for all manufacturers beginning with A), medium, or a subset of cables.

For each cable listed, three items will be returned: manufacturer, cable name, and medium (COAX or WAV).

A wildcard character (*) can be used as the last character of any string. Using the wildcard character should be read as 'any manufacturers starting with the characters before the wildcard'. Thus specifying "FR*" means any manufacturers starting "FR". The wildcard character can be used on its own where it means 'all manufacturers'. This also applies to the cable subset.

Example: `:FLOC:DBAS:CABL? "*", ANY, "*"`

List all the cables on the fault location database

Example response `Andrew, LDF5-50A, COAX, Andrew, LDF6-50, COAX, Andrew, LDF7-50A, COAX, ...`

:FLOCation
:DBASe
:CABLes
:NUMBer?

Parameters: <STRING PROGRAM DATA>, <CPD>, <STRING PROGRAM DATA>

manufacturer subset, medium subset, cable subset

Valid values: manufacturer subset: string. Maximum length of 32 characters excluding quotes. Excess characters will be ignored.

medium subset: [COAX | WAVeguide | ANY]. Values other than those stated are rejected and an error generated.

cable subset: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

number of cables

Returned values: number of cables: integer

Description: This command is to be used in conjunction with :FLOC:DBAS:CABL?. If the same parameters are supplied to this command as will be supplied to :FLOC:DBAS:CABL? then this command determines how many cables will be returned by the :FLOC:DBAS:CABL? command. This can be useful in reserving space for the :FLOC:DBAS:CABL? response.

Example: `:FLOC:DBAS:CABL:NUMB? "*", ANY, "*"`

Determine how many cables are in the database.

:FLOCation :DBASe :COPY [:DISK]

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

manufacturer, cable, file name

Valid values: manufacturer: string. Maximum length of 32 characters excluding quotes. Excess characters will be ignored.

cable: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Copy the data for the specified cable (using manufacturer and cable) from the fault location cable database and store it in a fault location cable store as specified by file name. An extension is not specified as it is fixed by the instrument. The cable database disk must be present. The manufacturer and cable are case independent. Fault location cable stores can only be stored in internal memory, not on the floppy disk. A fault location cable store can be copied to a floppy disk using :MMEM:COPY but the data will not be directly accessible from the floppy disk. It will be necessary to first copy the data back to a fault location cable store in the instrument.

Example: `:FLOC:DBAS:COPY "andrew", "ldf2-50", "and_ldf2"`

Copy the cable data for Andrew LDF2-50 into cable store AND_LDF2

:FLOCation :DBASe :ID [:DISK]?

Parameters: none

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the id string for the fault location cable database. The id string will contain version information for the cable database. The cable database disk must be present.

Example: `:FLOC:DBAS:ID?`

:FLOCation :DBASe :ID :STORe?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the user entered id string for the specified fault location cable store.

Example: `:FLOC:DBAS:ID:STOR? "FLEX12"`

Read the id string for the fault location cable store FLEX12.

:FLOCation :DBASe :MANufacturer [:LIST]?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first manufacturer, ..., last manufacturer

Returned values: first manufacturer: string. Maximum length of 32 characters excluding quotes.
...
last manufacturer: string. Maximum length of 32 characters excluding quotes.

Description: List all manufacturers on the fault location cable database. The cable database disk must be present.

Example: `:FLOC:DBAS:MAN?`

List all the manufacturers on the fault location cable database

:FLOCation
:DBASe
:MANufacturer
:NUMBer?

Parameters: none

Response: <NR1>

number of manufacturers

Returned values: number of manufacturers: integer

Description: This command is to be used in conjunction with :FLOC:DBAS:MAN. This command determines how many manufacturers will be returned by the :FLOC:DBAS:MAN? command. This can be useful in reserving space for the :FLOC:DBAS:MAN? response.

Example:
```
:FLOC:DBAS:MAN:NUMB?
```

Determine how many manufacturers are on the fault location database

:FLOCation
:DBASe
:READ
[:DISK]?

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

manufacturer, cable

Valid values: manufacturer: string. Maximum length of 32 characters excluding quotes. Excess characters will be ignored.

cable: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

Response: <CRD>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <NR2>, <STRING RESPONSE DATA>, <STRING RESPONSE DATA>

medium, cutoff freq/relative velocity, first freq, first attn, f2, a2, f3, a3, f4, a4, f5, a5, f6, a6, f7, a7, f8, a8, f9, a9, f10, a10, f11, a11, f12, a12, f13, a13, f14, a14, f15, a15, f16, a16, f17, a17, f18, a18, f19, a19, last freq, last attn, manufacturer name, cable name

Returned values: medium: [COAX | WAV]
cutoff freq/relative velocity: real

first freq: real
first attn: real
f2:real
a2:real
f3:real
a3:real
f4:real
a4:real
f5:real
a5:real
f6:real
a6:real
f7:real
a7:real
f8:real
a8:real
f9:real
a9:real
f10:real
a10:real
f11:real
a11:real
f12:real
a12:real
f13:real
a13:real
f14:real
a14:real

f15:real
a15:real
f16:real
a16:real
f17:real
a17:real
f18:real
a18:real
f19:real
a19:real
last freq: real
last attn: real
manufacturer string: string. Maximum length of 32 characters excluding quotes.
cable string: string. Maximum length of 20 characters excluding quotes.

Description: Return the data for the specified cable (using manufacturer and cable) from the fault location cable database. The cable database disk must be present. The manufacturer and cable are case independent. The frequency/attenuation table contains twenty pairs of reals. If the table contains less than twenty entries then the first entry with a frequency of zero terminates the table. Note that all twenty pairs of reals will still be returned regardless of the number of valid entries.

Example:

```
:FLOC:DBAS:READ? "andrew", "ldf2-50"
```

View the cable data for Andrew LDF2-50.

:FLOCation
:DBASe
:READ
:STORe?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <CRD>, <NR2>, <NR2>,

f16:real
a16:real
f17:real
a17:real
f18:real
a18:real
f19:real
a19:real
last freq: real
last attn: real

Description: Return the data in the specified fault location cable store. The frequency/attenuation table contains twenty pairs of reals. If the table contains less than twenty entries then the first entry with a frequency of zero terminates the table. Note that all twenty pairs of reals will still be returned regardless of the number of valid entries.

Example:
```
:FLOC:DBAS:READ:STOR? "my_cable"
```

View the cable data from cable store MY_CABLE.

:FLOCation :DBASe :SELect [:DISK]

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

manufacturer, cable

Valid values: manufacturer: string. Maximum length of 32 characters excluding quotes. Excess characters will be ignored.

cable: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

Description: Select the data for the specified cable (using manufacturer and cable) from the fault location cable database to be the data that the instrument uses when the cable database is turned on. The cable database disk must be present. The manufacturer and cable are case independent.

Once the data has been loaded, the database can be turned on using :FLOC:DBAS:STAT.

Example:
```
:FLOC:DBAS:SEL "andrew", "ldf2-50"; STAT ON
```

Select the cable data for Andrew LDF2-50 and turn on the database.

:FLOCation :DBASe :SELect :STORe

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Select the data from the specified fault location cable store to be the data that the instrument uses when the cable database is turned on.

Once the data has been loaded, the database can be turned on using :FLOC:DBAS:STAT.

Example: `:FLOC:DBAS:SEL:STOR "and_ldf2"; STAT ON`

Select the cable data from cable store AND_LDF2 *and turn on the database.*

:FLOCation :DBASe [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable fault location cable database.

Note that while the database is enabled, the user will not be able to set cutoff frequency, relative velocity or attenuation since these will be set by the instrument using data from the cable that was selected from the fault location cable database or a fault location cable store.
When disabling the database, the attenuation, cutoff frequency and relative velocity will remain at the values they were set to just before the database was disabled.

Example: `:FLOC:DBAS ON`

Enable fault location database.

:FLOCation :DBASe [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether cable database is enabled.

Example: `:FLOC:DBAS?`

:FLOCation
:DBASe
:WRITe
:STORe

Parameters: <STRING PROGRAM DATA>, <CPD>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>,<NRf>, <NRf>, <NRf>, <NRf>, <NRf>,<NRf>, <NRf>, <NRf>, <NRf>,<NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>, <NRf>,

f16: real
a16: real
f17: real
a17: real
f18: real
a18: real
f19: real
a19: real
f20: real
a20: real
manufacturer string: string. Maximum length of 32 characters excluding quotes. Excess characters will be ignored.
cable string: string. Maximum length of 20 characters excluding quotes. Excess characters will be ignored.

Description: Write the data to the specified fault location cable store. The frequency/attenuation table contains twenty pairs of reals. All twenty pairs must be sent. If less than twenty entries are required in the frequency/attenuation table then it is necessary to terminate the list with a frequency of zero. The easiest method is to send the whole of the remaining items in the table as 0. The frequencies must be in ascending order. A fault location cable store can be copied to a floppy disk using :MMEM:COPY but the data will not be directly accessible from the floppy disk. It will be necessary to first copy the data back to a fault location cable store in the instrument.

Example:
```
:FLOC:DBAS:WRIT:STOR "my_cable", ...etc
```
Write the cable data to the cable store MY_CABLE.

:FLOCation :ENTRy

Parameters: <CPD>

entry mode

Valid values: entry mode : [FREQuency | RANGe]. Values other than those stated are rejected and an error generated.

Description: Set entry mode to frequency or range entry.

In range entry mode, the range set using :FLOC:RANG will be used to determine the frequency span over which the source is swept. The start and stop frequencies can be read using the :MEAS:DOM:STAR? and :MEAS:DOM:STOP? commands.

In frequency entry mode, the start and stop frequencies can be set using :MEAS:DOM:STAR and :MEAS:DOM:STOP. The range will be calculated from these values and can be read using :FLOC:RANG?.

Example:
```
:FLOC:ENTRY RANG
```
Enable range entry mode.

:FLOCation :ENTRy?

Parameters: none

Response: <CRD>

entry mode

Returned values: entry mode : [FREQ | RANG]

Description: Determine whether entry mode is frequency or range entry.

Example: `:FLOC:ENTR?`

:FLOCation :MCORrection

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable masking correction.

Example: `:FLOC:MCOR ON`

Enable masking correction.

:FLOCation :MCORrection?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether masking correction is enabled for the active channel.

Example: `:FLOC:MCOR?`

:FLOCation :MEDium

Parameters: <CPD>

medium

Valid values: medium : [COAX | WAVeguide]. Values other than those stated are rejected and an error generated.

Description: Set medium of transmission line under test.

Example: `:FLOC:MED COAX`

Inform instrument that measurements will be made on a coax transmission line.

:FLOCation :MEDium?

Parameters: none

Response: <CRD>

medium

Returned values: medium: [COAX | WAV]

Description: Determine medium of transmission line under test.

Example:
```
:FLOC:MED?
```

:FLOCation :RANGe

Parameters: <NUMERIC VALUE>

range

Valid values: range: real

Suffix: range: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft

Description: Sets the range of the fault location system.

This command will only be accepted if range entry mode is enabled.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:FLOC:RANG 100
```

Set the fault location range to 100 m.

:FLOCation :RANGe?

Parameters: none

Response: <NR2>

range

Returned values: range: real

Description: Read the range of the fault location system. The value will either be returned in units of m or ft depending on the setting of :FLOC:UNIT.

Example:
```
:FLOC:RANG?
```

:FLOCation :RVELocity

Parameters: <NUMERIC VALUE>

relative velocity

Valid values: relative velocity: real

Suffix: relative velocity: No suffix is allowed.

Description: Set the relative velocity if the currently selected medium is coax. For waveguide medium an error will be generated.

This command will generate an error if the fault location cable database is enabled.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:FLOC:RVEL 0.8`

Set the relative velocity to 0.8.

:FLOCation :RVELocity?

Parameters: none

Response: <NR2>

relative velocity

Returned values: relative velocity: real

Description: Read the relative velocity if the currently selected medium is coax for the active channel. For waveguide medium an error will be generated. This command will always return the current value regardless of whether it was set by the user or the fault location cable database.

Example: `:FLOC:RVEL?`

:FLOCation :UNITs

Parameters: <CPD>

distance units

Valid values: distance units : [FEET | METRes]. Values other than those stated are rejected and an error generated.

Description: Set the distance units to feet or metres.

Example: `:FLOC:UNIT METR`

Set the distance units to metres.

:FLOCation :UNITs?

Parameters: none

Response: <CRD>

distance units

Returned values: distance units: [FEET | METR]

Description: Determine if distance units are set to feet or metres.

Example: `:FLOC:UNIT?`

:FLOCation :WGCutoff

Parameters: <NUMERIC VALUE>

waveguide cutoff frequency

Valid values: waveguide cutoff frequency: real

Suffix: waveguide cutoff frequency: The following suffixes are accepted for frequency: Hz, kHz, MHz or GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the waveguide cutoff frequency for the active channel.

This command will generate an error if the fault location cable database is enabled.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:FLOC:WGC 2.078E9`

Set the waveguide cutoff frequency to 2.078 GHz.

:FLOCation :WGCutoff?

Parameters: none

Response: <NR2>

waveguide cutoff frequency

Returned values: waveguide cutoff frequency: real

Description: Read the waveguide cutoff frequency. This command will always return the current value regardless of whether it was set by the user or the fault location cable database. The value is returned in units of Hz.

Example: `:FLOC:WGC?`

:FLOCation :WINDow

Parameters: <CPD>

window level

Valid values: window level : [LOW | MEDium | HIGH]. Values other than those stated are rejected and an error generated.

Description: Set the window level.

Example: `:FLOC:WIND MED`

Set the window level to medium.

:FLOCation :WINDow?

Parameters: none

Response: <CRD>

window level

Returned values: window level: [LOW | MED | HIGH]

Description: Read the window level.

Example: `:FLOC:WIND?`

:FLOCation :ZOOM

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable fault location zoom mode for the active channel. Zoom mode uses a chirp-z transform which allows a sub-range to be selected.

When zoom mode is enabled, the following commands will allow the display to show a subset of the range:
:DISP:CENT, :DISP:SPAN, :DISP:STAR, and :DISP:STOP.

Also, :DISP:CSP can be used to select either a start/stop display or a centre/span display.

When zoom mode is off, the display is always start/stop, the start is 0m or 0ft and the stop is the range.

Example: `:FLOC:ZOOM ON`

Enable zoomed mode.

:FLOCation :ZOOM?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether fault location zoom mode is enabled.

Example: `:FLOC:ZOOM?`

HARDcopy subsystem

```
HARDcopy
        ABORt
        BUILd
        CONFigure
                COPies\?
                DATE\?
                GANNotation\?
                GRATicule\?
                ID\?
                LIMits\?
                MANNotation\?
                MANufacturer\?
                MARKers\?
                MEASurement\?
                MTABle\?
                MTITle\?
                RECall
                SAVE
                SCONditions\?
                STITle\?
                TEXT
                        [STATe]\?
                        STRing\?
                TYPE\?
        DEVice
                ADDRess\?
                CURRent?
                DISK
                        INSTall
                        LIST?
                        NUMBer?
                LIST?
                NUMBer?
                REMove
                [SELect]
        LIMit
                CURRent
                [STORe]
        OUTPut
        [PLOT]
                [ALL]
                TRACes
        PORT\?
        SDUMp
        SETTings
                CURRent
                [STORe]
        TEST
```

:HARDcopy :ABORt

Parameters: none

Description: Abort the current hardcopy output. Note that any data already in the printer's buffer will still be printed.

Example:
```
:HARD:ABOR
```

:HARDcopy :BUILd

Parameters: none

Description: Start a print of the instrument build state on the currently selected output device.

Example:
```
:HARD:BUIL
```

:HARDcopy :CONFigure :COPies

Parameters: <NRf>

no of copies

Valid values: no of copies: integer

Description: Select how many copies of hardcopy output to produce. This value will apply to all future hardcopy operations. This applies to all types of hardcopy output. The value will be ignored if the output is being sent to a file.

Example:
```
:HARD:CONF:COP 5;:HARD:LIM:CURR
```

Print five copies of the limit line store associated with active trace.

:HARDcopy :CONFigure :COPies?

Parameters: none

Response: <NR1>

no of copies

Returned values: no of copies: integer

Description: Determine how many copies of each hardcopy output will be produced.

Example:
```
:HARD:CONF:COP?
```

:HARDcopy :CONFigure :DATE

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of date and time on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:DATE ON`

Plot date and time.

:HARDcopy :CONFigure :DATE?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of date and time is enabled.

Example: `:HARD:CONF:DATE?`

:HARDcopy :CONFigure :GANNotation

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of the graticule annotation on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:GANN ON`

Plot graticule annotation.

:HARDcopy :CONFigure :GANNotation?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of the graticule annotation is enabled.

Example: `:HARD:CONF:GANN?`

:HARDcopy
:CONFigure
:GRATicule

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of graticules on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:GRAT ON`

Plot graticules.

:HARDcopy
:CONFigure
:GRATicule?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of graticules is enabled.

Example: `:HARD:CONF:GRAT?`

:HARDcopy :CONFigure :ID

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name, user id

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

user id: string

Description: Set the user entered id string of the specified hardcopy settings store.

Example:
```
:HARd:CONF:ID "mark03", "My hardcopy settings"
```

:HARDcopy :CONFigure :ID?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the user entered id string of the specified hardcopy settings store. The id string gives additional details of the hardcopy settings store over what can be determined from the file name alone.

Example:
```
:HARD:CONF:ID? "mine"
```

:HARDcopy :CONFigure :LIMits

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of limit lines on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:LIM ON`

Plot limit lines.

:HARDcopy :CONFigure :LIMits?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of limit lines is enabled.

Example: `:HARD:CONF:LIM?`

:HARDcopy :CONFigure :MANNotation

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of the measurement annotation on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:MANN ON`

Plot measurement annotation.

:HARDcopy :CONFigure :MANNotation?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of the measurement annotation is enabled.

Example: `:HARD:CONF:MANN?`

:HARDcopy :CONFigure :MANufacturer

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of the manufacturers name on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:MAN ON`

Plot manufacturer's name.

:HARDcopy :CONFigure :MANufacturer?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of the manufacturer's name is enabled.

Example: `:HARD:CONF:MAN?`

:HARDcopy :CONFigure :MARKers

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of markers on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:MARK ON`

Plot markers.

:HARDcopy :CONFigure :MARKers?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of markers is enabled.

Example: `:HARD:CONF:MARK?`

:HARDcopy :CONFigure :MEASurement

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of measurements on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:MEAS ON`

Plot measurements.

:HARDcopy :CONFigure :MEASurement?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of measurements is enabled.

Example: `:HARD:CONF:MEAS?`

:HARDcopy :CONFigure :MTABle

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of marker table on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:MTAB ON`

Plot marker table.

:HARDcopy :CONFigure :MTABle?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of marker table is enabled.

Example: `:HARD:CONF:MTAB?`

:HARDcopy
:CONFigure
:MTITle

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of measurement titles on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:MTIT ON`

Plot measurement titles.

:HARDcopy
:CONFigure
:MTITle?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of measurement titles is enabled.

Example: `:HARD:CONF:MTIT?`

:HARDcopy
:CONFigure
:RECall

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Recall the hardcopy settings from the specified hardcopy settings store.

The following are recalled:

Whether manufacturer identity is plotted or not
Whether the screen title is plotted or not
Whether measurement titles are plotted or not
Whether the date & time is plotted or not
Whether the setup conditions are plotted or not
Whether graticules are plotted or not
Whether graticule annotation is plotted or not
Whether measurements are plotted or not
Whether measurement annotation is plotted or not
Whether limit lines are plotted or not
Whether markers are plotted or not
Whether marker tables are plotted or not
Whether the user entered text is plotted or not
Whether the instrument type number is plotted or not
The user entered text
The number of copies to plot

Example: `:HARD:CONF:REC "MY_SETT"`

Recall the hardcopy settings from store MY_SETT.

:HARDcopy
:CONFigure
:SAVE

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Save the current hardcopy settings to the specified hardcopy settings store.

Example: `:HARD:CONF:SAVE "MY_SETT"`

Save the current hardcopy settings to store MY_SETT.

:HARDcopy :CONFigure :SCONditions

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of the setup conditions on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:SCON ON`

Plot setup conditions.

:HARDcopy :CONFigure :SCONditions?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of the setup conditions is enabled.

Example: `:HARD:CONF:SCON?`

:HARDcopy :CONFigure :STITle

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of screen title on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:STIT ON`

Plot screen title.

:HARDcopy :CONFigure :STITle?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of screen title is enabled.

Example: `:HARD:CONF:STIT?`

:HARDcopy :CONFigure :TEXT [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of user text on a graphical output. The value is ignored for textual output.

Example: `:HARD:CONF:TEXT ON`

Plot user text.

:HARDcopy :CONFigure :TEXT [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of user text is enabled.

Example: `:HARD:CONF:TEXT?`

:HARDcopy :CONFigure :TEXT :STRing

Parameters: <STRING PROGRAM DATA>

user string

Valid values: user string: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Send user text for hardcopy output. Use :HARD:CONF:TEXT to display the text on the next plot.

Example: `:HARD:CONF:TEXT:STR "Lowpass filter"`

Set user text to Lowpass filter.

:HARDcopy :CONFigure :TEXT :STRing?

Parameters: none

Response: <STRING RESPONSE DATA>

user string

Returned values: user string: string. Maximum length of 30 characters excluding quotes.

Description: Determine the user text to be used for hardcopy output. Use :HARD:CONF:TEXT:STAT? to determine whether the text will be output.

Example:
```
:HARD:CONF:TEXT:STR?
```

:HARDcopy :CONFigure :TYPE

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable plotting of instrument type number on a graphical output. The value is ignored for textual output.

Example:
```
:HARD:CONF:TYPE ON
```

Plot instrument type number.

:HARDcopy :CONFigure :TYPE?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether plotting of instrument type number is enabled.

Example:
```
:HARD:CONF:TYPE?
```

:HARDcopy :DEVice :ADDRess

Parameters: <NRf>

address

Valid values: address: integer. Valid values are 0 to 30. Values outside range are rejected and an error generated.

Description: Set the GPIB address that the instrument will use for controlling, for instance, an HPGL plotter.

Example: `:HARD:DEV:ADDR 5`

Set plotter address to 5.

:HARDcopy :DEVice :ADDRess?

Parameters: none

Response: <NR1>

address

Returned values: address: integer. Values are in the range 0 to 30.

Description: Determine the GPIB address that the instrument will use for controlling an HPGL plotter.

Example: `:HARD:DEV:ADDR?`

:HARDcopy :DEVice :CURRent?

Parameters: none

Response: <STRING RESPONSE DATA>

hardcopy device in use

Returned values: hardcopy device in use: string

Description: Determine which hardcopy device is in use. This string will be the same as that used to select the device using :HARD:DEV[:SEL].

Example: `:HARD:DEV:CURR?`

:HARDcopy :DEVice :DISK :INSTall

Parameters: <STRING PROGRAM DATA>

hardcopy device

Valid values: hardcopy device: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Install a new hardcopy device into the instrument. The hardcopy device string must be one of the strings read using :HARD:DEV:DISK:LIST? or an error will be generated. Once the new hardcopy device has been installed into the instrument it can be selected using :HARD:DEV:SEL.

Example:
```
:HARD:DEV:DISK:INST "EPSON FX SERIES"
```

Install Epson printer driver into the instrument.

:HARDcopy :DEVice :DISK :LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first hardcopy device, ..., last hardcopy device

Returned values: first hardcopy device: string. Maximum length of 30 characters excluding quotes.
...
last hardcopy device: string. Maximum length of 30 characters excluding quotes.

Description: List all hardcopy devices on the floppy disk that are available to be installed into the instrument. The disk containing the hardcopy device(s) to be installed must be present. To install one of the hardcopy devices listed, use :HARD:DEV:DISK:INST using one of the strings returned by this command.

Example:
```
:HARD:DEV:DISK:LIST?
```

List all the hardcopy devices on the floppy disk

:HARDcopy :DEVice :DISK :NUMBer?

Parameters: none

Response: <NR1>

number of hardcopy devices

Returned values: number of hardcopy devices: integer

Description: This command is to be used in conjunction with :HARD:DEV:DISK:LIST. This command determines how many hardcopy devices will be returned by the :HARD:DEV:DISK:LIST? command. This can be useful in reserving space for the :HARD:DEV:DISK:LIST? response.

Example:
```
:HARD:DEV:DISK:NUMB?
```
Determine how many hardcopy devices are on the floppy disk

:HARDcopy :DEVice :LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first hardcopy device, ..., last hardcopy device

Returned values: first hardcopy device: string. Maximum length of 30 characters excluding quotes.
...
last hardcopy device: string. Maximum length of 30 characters excluding quotes.

Description: List all hardcopy devices available in the instrument. A hardcopy device may then be selected using :HARD:DEV with one of the strings returned by this command.

Example:
```
:HARD:DEV:LIST?
```
List all the hardcopy devices available.

:HARDcopy :DEVice :NUMBer?

Parameters: none

Response: <NR1>

number of hardcopy devices

Returned values: number of hardcopy devices: integer

Description: This command is to be used in conjunction with :HARD:DEV:LIST. This command determines how many hardcopy devices will be returned by the :HARD:DEV:LIST? command. This can be useful in reserving space for the :HARD:DEV:LIST? response.

Example:
```
:HARD:DEV:NUMB?
```
Determine how many hardcopy devices are available in the instrument.

:HARDcopy :DEVice :REMove

Parameters: <STRING PROGRAM DATA>

hardcopy device

Valid values: hardcopy device: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Remove a hardcopy device from the instrument. The hardcopy device string must be one of the strings read using :HARD:DEV:LIST? or an error will be generated. It may be useful to remove hardcopy devices from the instrument if it is known that they will not be used since it will allow memory to be freed for storage of other items.

Example:
```
:HARD:DEV:REM "EPSON FX SERIES"
```
Remove EPSON printer driver from the instrument.

:HARDcopy :DEVice [:SELect]

Parameters: <STRING PROGRAM DATA>

hardcopy device

Valid values: hardcopy device: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Select a hardcopy device for displayed text. The hardcopy device string must be one of the strings read using :HARD:DEV:LIST? or an error will be generated. Selecting a hardcopy device will set the output port to the default for that device. Use :HARD:PORT to change the port if necessary after the device has been selected.

Example: `:HARD:DEV "EPSON FX SERIES"`

Select EPSON printer driver.

:HARDcopy :LIMit :CURRent

Parameters: none

Description: Start a print of the limit line in use on the active measurement.

Example: `:HARD:LIM:CURR`

:HARDcopy :LIMit [:STORe]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Start a print of the selected limit line store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "C";:HARD:LIM "2TO4"`

Start a printout of limit line store named 2TO4 *in internal memory.*

:HARDcopy
:OUTPut

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Output the previously stored hardcopy output to the currently selected port. An error will be generated if the port is set to FILE.

This command allows hardcopy that was generated and sent to FILE to be output to a printer/plotter that is connected to the instrument. Note that since the data is already stored in the correct format for a specified printer/plotter, it is up to the user to make sure that the device connected is the correct type.

Example:
```
:HARD:PORT PAR,"";OUTP "NBFILT01"
```

Output the previously 'printed' data to the printer on the parallel port.

:HARDcopy [:PLOT] [:ALL]

Parameters: none

Description: Start a graphical plot on the currently selected output device. Only those items that have been configured to be output will be produced on the plot. If the output device is connected to the IEEE488 GPIB interface then the instrument must be given control of the bus for the duration of the hardcopy output and will return the bus back to the controller upon completion. The hardcopy output is done in the background and for hardcopy devices not connected to the GPIB, *OPC or *OPC? can be used to determine when the hardcopy output has completed.

Example: Plotting to a GPIB plotter when the controller is also on the GPIB bus requires that the instrument must be given control of the GPIB for the duration of the plot. The recommended steps for plotting under GPIB control are as follows:

1. Use the common command *PCB to tell the instrument the address of the controller. The instrument needs to know this so that control can be passed back when the plot is complete.

```
*PCB 21
```

Inform the instrument that the controller is located at address 21.

2. Initiate the plot

```
:HARD
```

3. Wait for the instrument to request control of the GPIB. The RQC bit (bit 1) of the Standard Event Status Register is set when the instrument is ready to accept control. This register may be read using *ESR?

```
*ESR?
```

4. Pass control to the instrument. Refer to the GPIB controller's operating manual for details.

5. When the plot has completed the instrument will pass control back to the controller.

:HARDcopy [:PLOT] :TRACes

Parameters: none

Description: Plot out only traces onto the hardcopy device. All displayed traces will be plotted. This is really only of use on a plotter where traces can be reliably overlapped. See :HARD[:PLOT][:ALL] for more information.

Example:

```
:HARD:TRAC
```

:HARDcopy :PORT

Parameters: <CPD>, <STRING PROGRAM DATA>

port, file name

Valid values: port: [GPIB | PARallel | SERial | FILE]. Values other than those stated are rejected and an error generated.
file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Select the port for hardcopy output. Note that changing the hardcopy output device will set the port to the default for that device. So always select the output device before selecting the port it is on.
The file name is ignored unless the port is set to FILE. Selecting FILE allows a hardcopy to be performed and saved for later printing. The saved hardcopy file can either be printed from the instrument using :HARD:OUTP or copied to a floppy/read by the remote controller for printing externally to the instrument. Since this command sets the filename to save the hardcopy output to, this command must be sent before every hardcopy command that is required to output to a file for later printing. Otherwise the same file will be overwritten each time and only the last hardcopy output will be saved.

Example: `:HARD:PORT PAR, ""`

Select the parallel port for hardcopy output.

:HARDcopy :PORT?

Parameters: none

Response: <CRD>, <STRING RESPONSE DATA>

port, file name

Returned values: port: [GPIB | PAR | SER | FILE]

file name: string

Description: Determine the currently selected port for hardcopy output. For any port other than FILE, the filename will be returned as a null string.

Example: `:HARD:PORT?`

Example response: SER,""

:HARDcopy :SDUMp

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Perform a screen dump of the screen (at 640 x 480). The file will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument (.BMP).

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Upon importing the file into windows, it will be treated as 96 dpi giving an image size of 6.7 x 5 inches.

Example:

```
:HARD:SDUM 'DUMP1'
```

:HARDcopy :SETTings :CURRent

Parameters: none

Description: Start a print of the current settings in use by the instrument on the currently selected output device.

Example:

```
:HARD:SETT:CURR
```

:HARDcopy
:SETTings
[:STORe]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Start a print of the selected settings store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "C";:HARD:SETT "LPF_MEAS"`

Start a printout of settings store named LPF_MEAS *in internal memory.*

:HARDcopy
:TEST

Parameters: none

Description: Start a print of the self test results on the currently selected output device.

Example: `:HARD:TEST`

INPut subsystem

INPut
- **ACDetection\?**
- **DEVice?**
- **FCORrection\?**
- **LCORrection\?**
- **OFFSet\?**
- **ZERO**
  - **AUTO\?**
  - **[DETectors]**

:INPut
:ACDetection

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Select either AC or DC detection for all scalar inputs. Note that disabling AC detection (using :INP:ACD OFF) will enable DC detection.

Example: `:INP:ACD ON`

Enable AC detection.

:INPut
:ACDetection?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether AC detection is enabled for all scalar inputs.

Example: `:INP:ACD?`

:INPut
:DEVice?

Parameters: <CPD>

input id

Valid values: input id: [A | B | C]. Values other than those stated are rejected and an error generated.

Response: <STRING RESPONSE DATA>

device

Returned values: device: string

Description: Determine which device is connected to the specified input.

The following devices can be connected:

"EEPROM detector"	Any of the EEPROM detectors
"623X detector"	A 6230 series detector
"6240F fault locator"	Fault locator port of a 6240F 10 MHz to 20 GHz
"6240M fault locator"	Fault locator port of a 6240M 10 MHz to 20 GHz
"6241 fault locator"	Fault locator port of a 6241 10 MHz to 20 GHz
"6243 fault locator"	Fault locator port of a 6243F 10 MHz to 26.5 GHz
"624X return loss"	Return loss port of a 624X
"65XX test head"	Transmission line test head
"Autotester cable"	
"Voltage cable"	
"None"	No device is connected to specified input
"Unknown"	A non-supported device is connected

Example:
```
:INP:DEV? A
```
Determine device connected to input A

:INPut :FCORrection

Parameters: <CPD>,<BOOLEAN PROGRAM DATA>

input id, state

Valid values: input id: [A | B | C]. Values other than those stated are rejected and an error generated.

Description: Turn flatness correction on or off for the specified input.

Example: `:INP:FCOR B, ON`

Turn on flatness correction for input B

:INPut :FCORrection?

Parameters: <CPD>

input id

Valid values: input id: [A | B | C]. Values other than those stated are rejected and an error generated.

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if flatness correction is enabled for the specified input.

Example: `:INP:FCOR? C`

Determine whether flatness correction is enabled on input C.

:INPut :LCORrection

Parameters: <CPD>,<CPD>

input id, correction state

Valid values: input id: [A | B | C]. Values other than those stated are rejected and an error generated.

Correction state: [AUTO | OFF]. Values other than those stated are rejected and an error generated.

Description: Set detector linearity correction on or off for specified input. If automatic correction is selected, the detector type is automatically determined and the appropriate correction applied to give a power measurement. If correction is turned off, the detector measurement will be a voltage rather than power.

Example: `:INP:LCOR C, AUTO`

Enable autosensing for input C

:INPut
:LCORrection?

Parameters: <CPD>

input id

Valid values: input id: [A | B | C]. Values other than those stated are rejected and an error generated.

Response: <CRD>

correction state

Returned values: correction state: [AUTO | OFF]

Description: Read detector linearity correction state for specified input.

Example: `:INP:LCOR? A`

Determine autosensing for input A

:INPut
:OFFSet

Parameters: <CPD>,<NRf>

input id, offset value

Valid values: input id: [A | B | C | RX]. Values other than those stated are rejected and an error generated.

offset value: real

Description: Set offset to be applied to all measurements using specified input. An offset can be applied to any of the three scalar analyzer inputs or to the spectrum analyser input (the receiver). The offset value is in dB's.

Example: `:INP:OFFS C,10`

Offset all measurements from input C by +10 dB.

:INPut
:OFFSet?

Parameters: <CPD>

input id

Valid values input id: [A | B | C | RX]. Values other than those stated are rejected and an error generated.

Response: <NR2>

offset value

Returned values: offset value: real

Description: Determine offset to be applied to all measurements using specified input. The offset value is in dB's.

Example:
```
:INP:OFFS? A
```
Read offset assigned to input A.

:INPut
:ZERO
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable automatic zeroing for all scalar detectors.

Example: `:INP:ZERO:AUTO ON`

Switch detector autozero on.

:INPut
:ZERO
:AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether automatic autozero is enabled.

Example: `:INP:ZERO:AUTO?`

:INPut
:ZERO
[:DETectors]

Parameters: none

Description: Zero all scalar detectors

Example: `:INP:ZERO`

MARKer subsystem

- MARKer
  - ACTive
    - ASSign\?
    - DATA?
    - POSition\?
    - [STATe]\?
  - BANDwidth
    - CFDF\?
    - [DATA]?
    - RWINdow
    - TARGet\?
  - DELTa
    - FIXed\?
    - POSition\?
    - [STATe]\?
  - MAXimum
  - MINimum
  - NOISe\?
  - PEAK
    - ALL
    - [FIND]
    - LEFT
    - NEXT
    - RIGHt
    - THReshold
      - [STATe]\?
      - VALue\?
  - PKPK
    - BANDwidth\?
    - MODE\?
    - RANGe
      - CENTer\?
      - CENTre\?
      - CSPan\?
      - SPAN\?
      - STARt\?
      - [STATe]\?
      - STOP\?
    - [RESult]?
  - POSition\?
  - RCARrier\?
  - READout\?
  - SEARch
    - DIRection\?
    - [RESult]?
    - TARGet\?
  - SLOPe\?

[STATe]\?
TABLe
[STATe]\?
TRACking\?

:MARKer
:ACTive
:ASSign

Parameters: <NRf>

marker number

Valid values: marker number: integer. Valid values are 1 to 8. Values outside range are rejected and an error generated.

Description: Designate a marker as the active marker. If the marker is not on, it will be turned on. The domain value of the marker will be clipped if necessary so that the active marker always appears on the screen.

Example: `:MARK:ACT:ASS 4`

Make marker 4 the active marker.

:MARKer
:ACTive
:ASSign?

Parameters: none

Response: <NR1>

marker number

Returned values: marker number: integer. Values are in the range 1 to 8.

Description: Determine which marker is the active marker.

Example: `:MARK:ACT:ASS?`

:MARKer :ACTive :DATA?

Parameters: none

Response: <NR3>

active marker measurement

Returned values: active marker measurement: real

The response value returned should be interpreted according to the active measurement format:

Log Power	dB or dBm
Linear Power	Watts
Voltage	Volts
VSWR	Units
Delay	Seconds

Description: Read the measurement at the active marker position. If the active marker is off, an error will be generated.

Example:
```
:MARK:ACT:DATA?
```

:MARKer :ACTive :POSition

Parameters: <NUMERIC VALUE>

domain value

Valid values: domain value: real

The domain value will be interpreted according to the active channel domain.

Suffix: domain value: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft
Time	ms, s	s

Description: Set the active marker to a domain value. This will automatically turn on the active marker if it was off. The active marker cannot be positioned off the screen, the domain value will be clipped if necessary.

If no suffix is entered for a distance value then the default suffix used will be either m or ft depending on the setting of :FLOC:UNIT.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:MARK:ACT:POS 15E9
```

Set the active marker to 15 GHz.

:MARKer :ACTive :POSition?

Parameters: none

Response: <NR2>

domain value

Returned values: domain value: real

Description: Read the active marker domain value.

Example: `:MARK:ACT:POS?`

:MARKer :ACTive [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set the active marker on or off. Turning off the active marker will automatically turn off the delta marker if it is on. Since the active marker cannot be positioned off the screen, turning the active marker on/off is equivalent to displaying the marker on not.

Example: `:MARK:ACT ON`

Switch on the active marker.

:MARKer :ACTive [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if the active marker is on.

Example: `:MARK:ACT?`

:MARKer
:BANDwidth
:CFDF

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set the marker centre frequency/delta frequency (CF/DF) calculation on or off. When it is turned on, the :MARKer:BANDwidth[:DATA]? command returns the CF/DF ratio in addition to the bandwidth and centre frequency values.

Example:
```
:MARK:BAND:CFDF ON
```

Enable the CF/DF calculation.

:MARKer
:BANDwidth
:CFDF?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if the marker centre frequency/delta frequency (CF/DF) calculation is on or off.

Example:
```
:MARK:BAND:CFDF?
```

:MARKer :BANDwidth [:DATA]?

Parameters: none

Response: <NR2>, <NR2>,<NR2>

bandwidth, centre frequency, CF/DF

Returned values: bandwidth: real

centre frequency: real

CF/DF: real

Description: Initiate a bandwidth search and return the bandwidth and centre frequency. If the search fails, the returned values will be 0. If the marker centre frequency/delta frequency calculations is enabled (:MARKer:BANDwidth:CFDF command), the value of this ratio is returned in addition to the bandwidth and centre frequency values, otherwise a value of 0 is returned. Bandwidth and centre frequency values are returned in units of Hz.

This command is only available for scalar channels.

If the search is successful the two highest numbered markers (excluding the active marker) are placed at the n dB points.

The bandwidth search window that is shown upon completion of this command can be removed from the display using :MARK:BAND:RWIN. Whilst the window is displayed, the extra information can (optionally) be output to hardcopy.

Example: `:MARK:BAND?`

Perform a bandwidth measurement.

:MARKer :BANDwidth :RWINdow

Parameters: none

Description: Remove the bandwidth search window (that was put up by the MARK:BAND? command) from the display.

Example: `:MARK:BAND:RWIN`

Remove the window.

:MARKer :BANDwidth :TARGet

Parameters: <NRf>

target

Valid values: target: real

Description: Set the target for a marker bandwidth search. The target value is in dB.

Example: `:MARK:BAND:TARG -3`

Set up the search target for 3 dB bandwidth measurements.

:MARKer :BANDwidth :TARGet?

Parameters: none

Response: <NR2>

target

Returned values: target: real

Description: Read the target for a marker bandwidth search. The target value is in dB.

Example: `:MARK:BAND:TARG?`

:MARKer :DELTa :FIXed

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable the function that fixes the spacing between the delta and the active marker.
When enabled, the active and delta markers will move as one so that the X domain offset between them remains fixed.
This command will generate an error if either the active or delta markers are not displayed.

Example: `:MARK:DELT:FIX ON`

Make the active and delta markers track each other.

:MARKer :DELTa :FIXed?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the delta marker is fixed to the active marker.

Example: `:MARK:DELT:FIX?`

:MARKer :DELTa :POSition

Parameters: <NUMERIC VALUE>

domain value

Valid values: domain value: real

The domain value will be interpreted according to the active channel domain.

Suffix: domain value: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft
Time	ms, s	s

Description: Set the delta marker to a domain value. This will automatically turn on the delta marker if it was off. The delta marker cannot be positioned off the screen, the domain value will be clipped if necessary.

If no suffix is entered for a distance value then the default suffix used will be either m or ft depending on the setting of :FLOC:UNIT.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:DELT:POS 12E9`

Set the delta marker to 12 GHz.

:MARKer :DELTa :POSition?

Parameters: none

Response: <NR2>

domain value

Returned values: domain value: real

Description: Read the delta marker domain value.

Example: `:MARK:DELT:POS?`

:MARKer :DELTa [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set the delta marker on or off. The delta marker will be set to the same domain value as the active marker when it is turned on. Turning on the delta marker will automatically turn on the active marker if it is off.

Example: `:MARK:DELT ON`

Switch on the delta marker.

:MARKer :DELTa [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the delta marker is on.

Example: `:MARK:DELT?`

:MARKer :MAXimum

Parameters: none

Description: Place the active marker at the trace maximum. If the active marker is off, it is turned on. This command is invalid in a spectrum analyzer channel. Use MARK:PEAK[:FIND] instead.

Example: `:MARK:MAX`

:MARKer
:MINimum

Parameters: none

Description: Place the active marker at the trace minimum. If the active marker is off, it is turned on. This command will give an error on a spectrum analyzer channel.

Example:
```
:MARK:MIN
```

:MARKer
:NOISe

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Turn on/off the noise in 1Hz bandwidth calculation.

This command is only valid in spectrum analyzer channels.

Example:
```
:MARK:NOIS ON
```

Enable the CF/DF calculation.

:MARKer
:NOISe?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if the marker noise in 1Hz bandwidth calculation is on or off.

Example:
```
:MARK:NOIS?
```

:MARKer
:PEAK
:ALL

Parameters: none

Description: Find the eight highest peaks and identify them with markers. This command only works on spectrum analyzer channels. The active marker is placed at the highest peak and the other seven markers are placed at the next lowest peaks. All of the markers are turned on. The marker table is also turned on.

Example:
```
:MARK:PEAK:ALL
```

:MARKer
:PEAK
[:FIND]

Parameters: none

Description: Place the active marker at the highest peak on the response (the trace maximum). If the active marker is off, it is turned on. The action of this command is identical to :MARK:MAX.

This command will generate an error if the channel is not a spectrum analyzer channel.

Example: `:MARK:PEAK`

:MARKer
:PEAK
:LEFT

Parameters: none

Description: Place the active marker at the next peak to the left of its current position. This command is only valid for fault location channels. If the active marker is off, it is turned on.
Using :MARK:PEAK:THR:VAL and :MARK:PEAK:THR it is possible to limit the search to peaks above a set threshold only. If the threshold is not set, then all peaks shown on the display are valid.

Example: `:MARK:PEAK:LEFT`

:MARKer
:PEAK
:NEXT

Parameters: none

Description: Find the next highest peak (below the current active marker response) and identify it with the active marker. This command only works on spectrum analyzer channels. This command will return an error if the active marker is not on.

Example: `:MARK:PEAK:NEXT`

:MARKer :PEAK :RIGHt

Parameters: none

Description: Place the active marker at the next peak to the right of its current position. This command is only valid for fault location channels. If the active marker is off, it is turned on.
Using :MARK:PEAK:THR:VAL and :MARK:PEAK:THR it is possible to limit the search to peaks above a set threshold only. If the threshold is not set, then all peaks shown on the display are valid.

Example:
```
:MARK:PEAK:RIGH
```

:MARKer :PEAK :THReshold [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

threshold state

Description: This function alters the effect of :MARK:PEAK:LEFT and :MARK:PEAK:RIGH. If threshold state is enabled then only peaks above the threshold set by :MARK:PEAK:THRE:VAL will be found when searching left/right. If the threshold state is off then all peaks will be found.
This command is only valid for fault location channels.

Example:
```
:MARK:PEAK:THR:VAL 10dB;STAT ON
```

Only include peaks above 10dB when using left/right peak search.

:MARKer :PEAK :THReshold [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

threshold state

Description: Determine whether the threshold is active for peak left/right searches.

Example:
```
:MARK:PEAK:THR?
```

:MARKer
:PEAK
:THReshold
:VALue

Parameters: <NUMERIC VALUE>

threshold value

Valid values: threshold value: real

Suffix threshold value: No suffix is allowed.

Description: Set the threshold value for a left/right peak search. The target value is in dB. To enable the entered value use :MARK:PEAK:THR ON.
This command is only valid for fault location channels.

Example: `:MARK:PEAK:THR:VAL 10;STAT ON`

Only include peaks above 10dB when using left/right peak search.

:MARKer
:PEAK
:THReshold
:VALue?

Parameters: none

Response: <NR2>

threshold value

Returned values: threshold value: real

Description: Read the threshold value for marker left/right peak searches. The target value is in dB. This command is only valid for fault location channels.

Example: `:MARK:PEAK:THR:VAL?`

:MARKer
:PKPK
:BANDwidth

Parameters: <NUMERIC VALUE>

frequency value

Valid values: frequency value: real

Suffix: frequency value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel with the active measurement set to group delay, this command will set the bandwidth window to be used when peak to peak tracking is turned on and the pk-pk mode is set to DIFF (i.e. max and min).

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:PKPK:BAND 5E6`

Set the bandwidth to 5 MHz.

:MARKer
:PKPK
:BANDwidth?

Parameters: none

Response: <NR2>

frequency value

Returned values: frequency value: real

Description: Determine the marker pk-pk bandwidth value for the active channel. An error will be generated if the channel is not a scalar channel.

Example: `:MARK:PKPK:BAND?`

:MARKer
:PKPK
:MODE

Parameters: <CPD>

pk-pk mode

Valid values: pk-pk mode: [LINear | PARabolic | DIFFerence]. Values other than those stated are rejected and an error generated.

Description: For a group delay measurement (scalar channel), this sets the mode of the marker peak to peak tracking function. This has no effect on peak to peak search or tracking functions outside of group delay.

If 'difference' is selected, the active marker is placed at the max response and the delta marker at the min response within the specified bandwidth window.

If 'linear' is selected, the active marker is placed at the point with the largest first derivative value (i.e. the largest slope).

If 'parabolic' is selected, the active marker is placed at the point with the largest second derivative value.

Example: `:MARK:PKPK:MODE LIN`

Set group delay marker pk-pk tracking to track largest first derivative value.

:MARKer
:PKPK
:MODE?

Parameters: none

Response: <CRD>

pk-pk mode

Returned values: pk-pk mode: [LIN | PAR | DIFF]

Description: Determine marker pk-pk tracking mode for the active channel.

Example: `:MARK:PKPK:MODE?`

Determine mode.

:MARKer :PKPK :RANGe :CENTer

Parameters: <NUMERIC VALUE>

pk-pk centre value

Valid values: pk-pk centre value: real

Suffix: pk-pk centre value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the marker pk-pk sub-range centre domain value. This command is only valid on a scalar channel. This is used for both peak to peak search and tracking functions. If the sub-range is turned on, this command will set the centre frequency of the sub-range.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:PKPK:RANG:CENT 12.4E9`

Set pk-pk sub-range centre frequency to 12.4 GHz.

:MARKer :PKPK :RANGe :CENTer?

Parameters: none

Response: <NR2>

pk-pk centre value

Returned values: pk-pk centre value: real

Description: Read the marker pk-pk sub-range centre domain value.

Example: `:MARK:PKPK:RANG:CENT?`

:MARKer
:PKPK
:RANGe
:CENTre

Parameters: <NUMERIC VALUE>

pk-pk centre value

Valid values: pk-pk centre value: real

Suffix: pk-pk centre value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the marker pk-pk sub-range centre domain value. This command is only valid on a scalar channel. This is used for both peak to peak search and tracking functions. If the sub-range is turned on, this command will set the centre frequency of the sub-range.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:PKPK:RANG:CENT 12.4E9`

Set pk-pk sub-range centre frequency to 12.4 GHz.

:MARKer
:PKPK
:RANGe
:CENTre?

Parameters: none

Response: <NR2>

pk-pk centre value

Returned values: pk-pk centre value: real

Description: Read the marker pk-pk sub-range centre domain value.

Example: `:MARK:PKPK:RANG:CENT?`

:MARKer
:PKPK
:RANGe
:CSPan

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: This will select between MMI entry of start/stop or centre/span values for marker pk-pk sub-range. This command is unlikely to be used but has been included for completeness.

Valid for scalar channels only.

Example: `:MARK:PKPK:RANG:CSP ON`

Set MMI to enter centre / span mode of marker pk-pk sub-range entry.

:MARKer
:PKPK
:RANGe
:CSPan?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the entry of the marker pk-pk sub-range is in start / stop mode or centre / span mode.

Example: `:MARK:PKPK:RANG:CSP?`

Determine if in centre / span or start / stop mode.

:MARKer
:PKPK
:RANGe
:SPAN

Parameters: <NUMERIC VALUE>

pk-pk span value

Valid values: pk-pk span value: real

Suffix: pk-pk span value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the marker pk-pk sub-range span value. This command is valid for scalar channels only. This is used for both peak to peak search and tracking functions. If the sub-range is turned on, this command will set the span of the sub-range.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:PKPK:RANG:SPAN 10E6`

Set the frequency span to 10 MHz.

:MARKer
:PKPK
:RANGe
:SPAN?

Parameters: none

Response: <NR2>

pk-pk span value

Returned values: pk-pk span value: real

Description: Read the marker pk-pk sub-range span value.

Example: `:MARK:PKPK:RANG:SPAN?`

:MARKer :PKPK :RANGe :STARt

Parameters: <NUMERIC VALUE>

pk-pk start domain

Valid values: pk-pk start domain: real

Suffix: pk-pk start domain: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the marker pk-pk sub-range start value. This command is only valid for a scalar channel. This is used for both peak to peak search and tracking functions. If the sub-range is turned on, this command will set the start frequency of the sub-range.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:PKPK:RANG:STAR 1E9`

Set 1 GHz start frequency.

:MARKer :PKPK :RANGe :STARt?

Parameters: none

Response: <NR2>

pk-pk start domain

Returned values: pk-pk start domain: real

Description: Read the marker pk-pk sub-range start value.

Example: `:MARK:PKPK:RANG:STAR?`

:MARKer :PKPK :RANGe [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set marker pk-pk (or pk-pk tracking) sub-range on or off. If off, the whole display span is used for the search.

Example: `:MARK:PKPK:RANG ON`

Set marker pk-pk sub-range on.

:MARKer
:PKPK
:RANGe
[:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether marker pk-pk sub-range is on or off.

Example: `:MARK:PKPK:RANG?`

Determine whether marker pk-pk sub-range is on.

:MARKer
:PKPK
:RANGe
:STOP

Parameters: <NUMERIC VALUE>

pk-pk stop domain

Valid values: pk-pk stop domain: real

Suffix: pk-pk stop domain: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the marker pk-pk sub-range stop value. This command is only valid for a scalar channel. This is used for both peak to peak search and tracking functions. If the sub-range is turned on, this command will set the stop frequency of the sub-range.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:PKPK:RANG:STOP 22E9`

Set 22 GHz marker pk-pk sub-range stop frequency.

:MARKer :PKPK :RANGe :STOP?

Parameters: none

Response: <NR2>

pk-pk stop domain

Returned values: pk-pk stop domain: real

Description: Read the marker pk-pk sub-range stop value.

Example: `:MARK:PKPK:RANG:STOP?`

:MARKer :PKPK [:RESult]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>, <NR3>

search result, pk-pk measurement

Returned values: pk-pk measurement: real

Description: This command has two functions.

If peak-to-peak tracking is enabled, then issuing this command will just return the last search result. This works for all peak to peak tracking, including group delay (if the option is fitted). Note that this means that it is not possible to do a manual peak to peak search whilst peak to peak tracking is on.

If peak-to-peak tracking is off then the functionality is as follows:
Initiate a peak-peak measurement, returning the result. This command is only valid for scalar channels. The active marker is placed at the trace maximum and the delta marker at the trace minimum. The markers are turned on if necessary. A search result of 0 indicates that the search failed (no data to perform search on). A search value of 1 indicates that the search was successful and the pk-pk measurement is valid.

For group delay measurements the result is in units of:

Seconds	if performing difference (max/min) pk-pk tracking
Seconds/Hz	if deviation from linear (first derivative) selected
$Seconds/Hz^2$	if deviation from parabolic (second derivative) selected

Example: `:MARK:PKPK?`

:MARKer :POSition

Parameters: <NRf>, <NUMERIC VALUE>

marker number, domain value

Valid values: marker number: integer. Valid values are 1 to 8. Values outside range are rejected and an error generated.

domain value: real

The domain value will be interpreted according to the active channel domain.

Suffix: domain value: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Frequency	Hz, kHz, MHz, GHz	Hz
Power	dBm	dBm
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft
Time	ms, s	s

Description: Set a marker to a domain value. This will automatically turn on the marker if it was off.

If no suffix is entered for a distance value then the default suffix used will be either m or ft depending on the setting of :FLOC:UNIT.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:MARK:POS 7,10E9`

Place marker 7 at 10 GHz.

:MARKer :POSition?

Parameters: <NRf>

marker number

Response: <NR2>

domain value

Returned values: domain value: real

Description: Read domain value of specified marker.

Example: `:MARK:POS? 6`

Read position of marker 6

:MARKer :RCARrier

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enabled/ disables relative to carrier mode. Only valid for spectrum analyzer channels.

Enabling this will make the spectrum analyzer measurement displayed relative to a single reading. That reading is the response at the active marker at the point when the relative to carrier is enabled. The effect of this is to make the value at the active marker 0 dB. The 0 dB position will also be moved to the top of the screen. The format will be changed to dBc.

Example: `:MARK:RCAR ON`

Enable relative to carrier mode.

:MARKer :RCARrier?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if the relative to carrier mode is enabled.

Example: `:MARK:RCAR?`

:MARKer :READout

Parameters: <CPD>

readout size

Valid values: readout size: [LARGe | NORMal]. Values other than those stated are rejected and an error generated.

Description: Set the marker readout size on the active trace.

Example: `:MARK:READ LARG`

Set the large readout mode for the active trace.

:MARKer :READout?

Parameters: none

Response: <CRD>

readout size

Returned values: readout size: [LARG | NORM]

Description: Determine the marker readout size on the active trace.

Example: `:MARK:READ?`

:MARKer :SEARch :DIRection

Parameters: <CPD>

search direction

Valid values: search direction: [LEFT | RIGHt]. Values other than those stated are rejected and an error generated.

Description: Set the direction for marker search operations.

Example: `:MARK:SEAR:DIR RIGH;TARG -10;RES?`

Perform a marker search for -10dB to the right of the current point.

:MARKer :SEARch :DIRection?

Parameters: none

Response: <CRD>

search direction

Returned values: search direction: [LEFT | RIGH]

Description: Determine the direction for marker search operations.

Example: `:MARK:SEAR:DIR?`

:MARKer :SEARch [:RESult]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

target found flag

Description: Initiate a marker search and return the result as a success or failure. The active marker is moved if the search was successful. Use :MARK:ACT:POS? and :MARK:ACT:DATA? to determine the result of the search.

Example:
```
:MARK:SEAR?
```
Perform a marker search using previously entered direction and target.

:MARKer :SEARch :TARGet

Parameters: <NRf>

target response

Valid values: target response: real

The response value will be interpreted according to the active channel format:

Log Power	dB or dBm
Linear Power	Watts
Voltage	V
VSWR	Units
Delay	Seconds

Description: Set the search target value for marker search operations. If the delta marker is on then the search target will be relative to the delta marker response value.

Example:
```
:MARK:SEAR:TARG -3.0
```
Set the search target to –3.0 dB.

:MARKer :SEARch :TARGet?

Parameters: none

Response: <NR3>

target response

Returned values: target response: real

Description: Determine the search target value for marker search operations.

Example:
```
:MARK:SEAR:TARG?
```

:MARKer :SLOPe

Parameters: <CPD>

marker slope

Valid values: marker slope: [DBOCtave | DBDecade | OFF]. Values other than those stated are rejected and an error generated.

Description: Set the marker slope function for the active measurement:

DBOCtave	dB/octave
DBDecade	dB/decade
OFF	Disable marker slope function

When marker slope function is enabled the marker readout value displayed is the slope of the response at the active marker. Each measurement can be set independently.

This command is only valid when:
- Channel is a scalar channel
- The domain is frequency
- Start frequency is not equal to stop frequency
- The format is log (dB)
- The delta marker is off.

Example: `:MARK:SLOP DBD`

Set the marker slope function to dB/decade on the active measurement.

:MARKer :SLOPe?

Parameters: none

Response: <CRD>

marker slope

Returned values: marker slope: [DBOC | DBD | OFF]

Description: Determine the marker slope function for the active measurement.

Example: `:MARK:SLOP?`

:MARKer [:STATe]

Parameters: <NRf>, <BOOLEAN PROGRAM DATA>

marker number, state

Valid values: marker number: integer. Valid values are 1 to 8. Values outside range are rejected and an error generated.

Description: Set a marker on or off. Turning the active or delta marker on will clip the marker if necessary so that it is always displayed. All other markers may not be displayed even if they are turned on as their domain value may be outside the screen domain values.

Example: `:MARK 4,ON`

Set marker 4 on.

:MARKer [:STATe]?

Parameters: <NRf>

marker number

Valid values: marker number: integer. Valid values are 1 to 8. Values outside range are rejected and an error generated.

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether a marker is on or off.

Example: `:MARK? 5`

Determine whether marker 5 is on.

:MARKer :TABLe [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

marker table state

Description: Turn marker table on or off. If the marker table is on, it will be displayed on the screen and output on hard copy. The displayed marker table applies per channel.

Example: `:MARK:TABL ON`

Display the marker table on the active channel.

:MARKer
:TABLe
[:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

marker table state

Description: Determine whether marker table is on or off for the active channel.

Example:
```
:MARK:TABL?
```

:MARKer
:TRACking

Parameters: <CPD>

tracking

Valid values: tracking: [OFF | MAX | MIN | PEAK | PKPK | BWSearch]. Values other than those stated are rejected and an error generated.

Description: Select the required marker tracking (or none). If tracking is enabled, the specified marker function is performed at the end of the measurement update. Marker tracking functions are set per measurement, but since markers are coupled between the two traces in a channel the function appears to be applied per channel. Thus if measurement one is set up to track maximum and measurement two is set up to track minimum, changing the active measurement from measurement one to measurement two will change the marker tracking from tracking maximum to tracking minimum. Note also that markers are coupled across channels unless the channels have different setups. Thus changing the active channel will change the tracking to that set up for the active measurement in the new channel.

The following functions have a tracking feature:

MAX	Track maximum. The marker will move to the maximum response at the end of each measurement update. Only valid for scalar and fault location channels.
MIN	Track minimum. The marker will move to the minimum response at the end of each measurement update. Only valid for scalar and fault location channels.
PEAK	Track peak value. The marker will move to the maximum response at the end of each measurement update. Only valid for spectrum analyzer channels.
PKPK	Track peak-to-peak. Active marker will move to maximum response and delta marker to minimum response at the end of each measurement update, except for group delay when calculating deviation from linear or parabolic when only the active marker is moved. This is valid for scalar channels only. When performing difference (max and min) pk-pk tracking on a group delay measurement, the search is performed using the bandwidth window set using MARK:PKPK:BAND. To read the peak to peak tracking result, use MARK:PKPK?
BWSearch	Track bandwidth search. A bandwidth search about the active marker will be performed at the end of each measurement update. This is valid on scalar channels only.

Example: `:MARK:TRAC PKPK`

Set pk-pk marker tracking.

:MARKer
:TRACking?

Parameters: none

Response: <CRD>

tracking

Returned values: tracking: [OFF | MAX | MIN | PEAK | PKPK | BWS]

Description: Determine the marker tracking state for the active trace.

Example:
```
:MARK:TRAC?
```

MEASurement subsystem

```
MEASurement
    ACTive\?
    AVERaging
        NUMBer\?
        RESTart
        [STATe]\?
    [DATA]
        [ASCii]?
        BINary?
        POINts?
    FORMat\?
    HOLD\?
    LIMit
        CHECk\?
        ID\?
        NSEGments?
        OFFSet\?
        RESet
        SAVE
        SEGMent\?
        SELect
        [STATe]\?
    MEASure
        DEFinition?
        LIVE
        MEMory
            [BINary]
                [ONLY]
                SETTings
            ID\?
        POWer
        RATio
    MOPeration
        SELect
            [BINary]
        [STATe]\?
    NMEas\?
    SAVE
        [BINary]
        TEXT
    STATe\?
```

:MEASurement :ACTive

Parameters: <NRf>

measurement number

Valid values: measurement number: integer. Valid values are 1 to 2. Values outside range are rejected and an error generated.

Description: Set the active measurement within the active channel. Either measurement 1 or measurement 2 may be made active. The active measurement is always displayed. If only one measurement is currently being displayed and the other measurement is made active then the currently active measurement will be replaced by the other measurement.

Use :MEAS:NME to change the number of measurements displayed.

Example: `:MEAS:ACT 2`

Make measurement 2 active.

:MEASurement :ACTive?

Parameters: none

Response: <NR1>

measurement number

Returned values: measurement number: integer. Values are in the range 1 to 2.

Description: Determine the active measurement within the active channel.

Example: `:MEAS:ACT?`

:MEASurement :AVERaging :NUMBer

Parameters: <NUMERIC VALUE>

averaging number

Valid values: averaging number: integer

Suffix: averaging number: No suffix is allowed.

Description: Set the averaging number for the active measurement in the active channel. This command is only valid on scalar or fault location live measurements.

Use :MEAS:AVER to turn averaging on or off.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE. Using UP and DOWN will step the averaging number in a 1, 2, 4, 8, 16, ... sequence.

Example: `:MEAS:AVER:NUMB 16;STAT ON`

Set the averaging number to 16 and apply averaging.

:MEASurement :AVERaging :NUMBer?

Parameters: none

Response: <NR1>

averaging number

Returned values: averaging number: integer

Description: Determine the averaging number for the active measurement in the active channel. This command is only valid on scalar or fault location live measurements.

Example: `:MEAS:AVER:NUMB?`

:MEASurement :AVERaging :RESTart

Parameters: none

Description: Restart averaging for active measurement in active channel. This command is only valid on scalar or fault location live measurements.

Example: `:MEAS:AVER:REST`

:MEASurement :AVERaging [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set the averaging for the active measurement in the active channel on or off. This command is only valid on scalar or fault location live measurements.

Example: `:MEAS:AVER ON`

Enable averaging.

:MEASurement :AVERaging [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether averaging for the active measurement in the active channel is on or off. . This command is only valid on scalar or fault location live measurements.

Example: `:MEAS:AVER?`

:MEASurement
[:DATA]
[:ASCii]?

Parameters: none

Response: <NR3>,...,<NR3>

response at point 0, ..., response at point *n*-1

Returned values: response at point 0: real
...
response at point *n*-1: real

Description: Read measurement data of the active measurement in ASCII format. Only those points displayed on the screen are returned. Units of data returned depends on measurement format:

Log Power	dB or dBm
Linear Power	Watts
Voltage	V
VSWR	Units
Delay	Seconds (Group delay option)

Note that for a spectrum analyzer measurement that is not in demodulation mode then the data will be returned as pairs of values (a minimum and a maximum with the minimum data being sent first).

Example: `:MEAS?`

Read measurement data.

Example response: -1.012763E+001,-1.009112E+001,-1.003542E+001,...,1.554673E-001,4.977528E-001

:MEASurement
[:DATA]
:BINary?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

measurement data

Description: Read measurement data of the active measurement in Binary format. Binary format allows faster transfer of measurement data. Only those points displayed on the screen are returned. Units of data returned depends on measurement format:

Log Power	dB or dBm
Linear Power	Watts
Voltage	V
VSWR	Units
Delay	Seconds (Group delay option)

The returned data is organised as follows:

Each measurement consists of 4 bytes received in the order:

byte 0 :	S E E E E E E E
byte 1 :	E F F F F F F F
byte 2 :	F F F F F F F F
byte 3 :	F F F F F F F F

Where S is sign bit, E is exponent (8 bits), and F is fractional part (23 bits). These bytes hold a 32 bit (IEEE single precision) number conforming to the IEEE Standard of Binary Floating-Point Arithmetic, ANSI/IEEE Std 754-1985.

Note that for a spectrum analyzer measurement that is not in demodulation mode then the data will be returned as pairs of values (a minimum and a maximum with the minimum data being sent first).

Example:
```
:MEAS:BIN?
```

:MEASurement
[:DATA]
:POINts?

Parameters: none

Response: <NR1>

displayed points

Returned values: displayed points: integer

Description: Read number of points displayed on the screen. Note that this returns the maximum number of data items that can be returned by :MEAS:DATA:ASC? or :MEAS:DATA:BIN?. This may be useful to grab enough storage space (e.g. malloc() in C) to store the response in.

Note that for a spectrum analyzer measurement that is not in demodulation mode then the data will be returned as pairs of values (a minimum and a maximum) This command takes account of this and will return the maximum number of reals that the MEAS:DATA:ASC? or MEAS:DATA:BIN? can return.

Example:
```
:MEAS:POIN?
```

:MEASurement :FORMat

Parameters: <CPD>

measurement format

Valid values: measurement format: [VSWR | LOG | WATT | VOLT | PCT | DBUV]. Values other than those stated are rejected and an error generated.

Description: Set the display format for the active measurement. Formats that are not applicable to the active measurement will generate an error. LOG format will be displayed as either dB or dBm depending on whether the measurement is absolute or relative.

On a scalar measurement measuring absolute power, the following formats are valid: LOG (dBm), WATT, and VOLT.
On a scalar measurement measuring relative power, the following formats are valid: LOG (dB), and VSWR.

On a fault location measurement, the following formats are valid: LOG (dB) and VSWR.

On spectrum analyzer measurements, in normal mode, measuring absolute power from the receiver, the following formats are valid: LOG (dBm), VOLT, and DBUV (dBµV).
On spectrum analyzer measurements, in normal mode, measuring relative power (i.e. RX/C or a normalised RX measurement), this command will give an error since the format at fixed at dB.
On FM demodulated spectrum analyzer measurements, this command will give an error since the format is fixed as frequency.
On group delay measurements, this command will give an error since the format is fixed as time (delay).

Example: `:MEAS:FORM VSWR`

Set VSWR format.

:MEASurement :FORMat?

Parameters: none

Response: <CRD>

current measurement format

Returned values: current measurement format: [VSWR | LOG | WATT | VOLT | PCT | DBUV | FREQ | TIME]

Description: Determine the display format for the active measurement.

Example: `:MEAS:FORM?`

:MEASurement :HOLD

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Hold active measurement. The active measurement is held immediately. If data is not valid for every point then the data for the last point is repeated as necessary. Thus a held measurement will always have valid data.

Example: `:MEAS:HOLD OFF`

Release held measurement.

:MEASurement :HOLD?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether active measurement is held.

Example: `:MEAS:HOLD?`

:MEASurement :LIMit :CHECk

Parameters: <CPD>

state

Valid values: state: [UPPer | LOWer | BOTH]. Values other than those stated are rejected and an error generated.

Description: Sets the limit specification associated with the active trace to be an upper only limit spec, a lower only limit spec or an upper and lower limit spec.

Example: `:MEAS:LIM:CHEC UPP`

Set the limit spec to be an upper only spec.

:MEASurement :LIMit :CHECk?

Parameters: none

Response: <CRD>

state

Returned values: state: [UPP | LOW | BOTH]

Description: Determine whether the limit specification associated with the active trace is an upper only limit spec, a lower only limit spec or an upper and lower limit spec.

Example: `:MEAS:LIM:CHEC?`

:MEASurement :LIMit :ID

Parameters: <STRING PROGRAM DATA>

id string

Valid values: id string: string. Maximum length of 256 characters excluding quotes. Excess characters will be ignored.

Description: Set the user description for the limit line on the active measurement.

Example: `:MEAS:LIM:ID "2-4G Flat lim at 5dB"`

:MEASurement :LIMit :ID?

Parameters: none

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string. Maximum length of 256 characters excluding quotes.

Description: Read the user entered id string for the limit line in use by the active trace.

Example: `:MEAS:LIM:ID?`

:MEASurement :LIMit :NSEGments?

Parameters: none

Response: <NR1>

number of segments

Returned values: number of segments: integer. Values are in the range 0 to 12.

Description: Read the number of segments in the limit specification associated with the active measurement.

Example:
```
:MEAS:LIM:NSEG?
```

:MEASurement :LIMit :OFFSet

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Sets the limit specification associated with the active trace to have domain values which are offsets from a centre value rather than actual domain values. A limit specification that is defined as an offset is symmetrical about the centre value and therefore only one half of the limit specification needs to be defined.

Example:
```
:MEAS:LIM:OFFS ON
```

Set the limit spec to be defined using offset domain values.

:MEASurement :LIMit :OFFSet?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the limit specification associated with the active trace is defined using domain offsets or actual domain values.

Example:
```
:MEAS:LIM:OFFS?
```

:MEASurement
:LIMit
:RESet

Parameters: none

Description: Reset the limit segment counter in the limit specification associated with the active trace to address the first limit segment. This command is used in conjunction with the :MEASurement:LIMit:SEGment command.

Example:
```
:MEAS:LIM:RES
```
Prepare specification for reading/writing limit segments.

:MEASurement
:LIMit
:SAVE

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Save the limit specification associated with the active trace to a limit specification store. The allows the limit specification to be recalled at a later date or to be applied to more than one trace. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:
```
:MMEM:MSIS "C";:MEAS:LIM:SAVE "HPF_2G"
```
Save limit spec to a file named HPF_2G *in internal memory.*

:MEASurement
:LIMit
:SEGMent

Parameters: <NRf>,<NRf>,<CPD>,<NRf>,<NRf>,<NRf>,<NRf>

start domain, stop domain, limit type, start upper, start lower, stop upper, stop lower

Valid values: start domain: real
stop domain: real
limit type: [FLAT | SLOPe]. Values other than those stated are rejected and an error generated.
start upper: real
start lower: real
stop upper: real
stop lower: real

When limit type is FLAT, only start upper and start lower will be used (The stop lower and stop upper parameters must be sent to conform with the command syntax, but their values will be ignored). The domain and limit values are stored as real numbers which are interpreted according to the active channel domain and the active measurement format respectively:

DOMAIN	UNITS	FORMAT	UNITS
Frequency	Hz	VSWR	Units
Power	dBm	Log Power	dB or dBm
Distance	m or ft	Linear Power	W
		Voltage	V
		Delay	S

Description: Write a segment to the selected limit specification. A limit specification consists of one or more limit segments.

To send a specification:

1. Send :MEAS:LIM:RES

This resets a counter that addresses the list of limit segments.

2. Send limit segments using :MEAS:LIM:SEGM

After each :MEAS:LIM:SEGM command is received, the counter increments automatically to address the next free segment.

The limit line is now created for the active trace. If the limit line specification is to be saved it should be done so using :MEAS:LIM:SAVE.

Example:
```
:MEAS:LIM:RES; SEGM 2E9, 8E9, FLAT, +0.5, -0.5, 0, 0;
 SEGM 8E9, 14E9, SLOPE, +0.5, -0.5, +1.0, -1.0
```

A limit specification is defined in two segments. Between 2 GHz and 8 GHz flat limit lines are placed at ±0.5 dB. Between 8 GHz and 14 GHz, a pair of sloped limit lines are introduced, starting at ±0.5 dB at 8 GHz and expanding to ±1.0 dB at 14 GHz.

:MEASurement
:LIMit
:SEGMent?

Parameters: none

Response: <NR2>,<NR2>,<CRD>,<NR3>,<NR3>,<NR3>,<NR3>

start domain, stop domain, limit type, start upper, start lower, stop upper, stop lower

Returned values:
start domain: real
stop domain: real
limit type: [FLAT | SLOP]
start upper: real
start lower: real
stop upper: real
stop lower: real

When limit type is FLAT, only start upper and start lower will be valid (The stop lower and stop upper parameters will be returned in accordance with the response syntax, but their values can be ignored).

Description: Read a limit segment from the currently selected limit store. A limit specification held in a store consists of one or more limit segments. To read a specification:

1. Send :MEAS:LIM:RES

This resets a counter that addresses the list of limit segments within the specification.

2. Send :MEAS:LIM:NSEG?

This returns the number of segments in the limit specification held in the current limit store.

3. Read limit segments using :MEAS:LIM:SEGM?

After each :MEAS:LIM:SEGM? command is received, the counter increments automatically to address the next segment. An error will be generated if an attempt is made to read beyond the last valid segment.

Example:
```
:MEAS:LIM:RES; NSEG?
```
Reset the store pointer and read the number of segments.

```
:MEAS:LIM:SEGM?
```
Read the first limit segment.

:MEASurement :LIMit :SELect

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Assign a limit specification to the active measurement. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if loading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MEAS:LIM:SEL "BPF_8"`

Assign limit specification BPF_8 *to the active measurement.*

:MEASurement :LIMit [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable limit checking. The limit specification used is that assigned to the active measurement. Note that limit checking is not applied to demodulated spectrum analyzer measurements.

Example: `:MEAS:LIM ON`

Switch limit checking on.

:MEASurement :LIMit [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether limit checking is enabled.

Example: `:MEAS:LIM?`

:MEASurement :MEASure :DEFinition?

Parameters: none

Response: <CRD>

measurement definition

Returned values: measurement definition: [A | B | C | AC | BC | RX | RXC | GDEL | MEM]

Description: Determine the measurement definition for the active measurement on the active channel:

A, B, or C	Absolute measurement using scalar input specified
AC	Ratio (A/C) measurement
BC	Ratio (B/C) measurement
RX	Absolute measurement using spectrum analyzer receiver
RXC	Ratio (RX/C) measurement
GDEL	Group delay measurement
MEM	Trace memory display

Note that for fault location channels, the user cannot set the inputs in use, they are set automatically by sensing the type of fault locator fitted to the instrument. If a 6420 series fault locator is attached then a single input will be selected, otherwise it is assumed that a 6581 or 6583 is attached and B over C will be selected.

Example:
```
:MEAS:MEAS:DEF?
```

:MEASurement :MEASure :LIVE

Parameters: none

Description: Set up a live measurement on the active measurement in the active channel.

For fault location channels, this command is only valid if the active measurement is measurement 1. The inputs used for the measurement are selected by the instrument depending on the type of fault locator attached. If a 6240 series fault locator is attached then a single input will be selected, otherwise it is assumed that a 6581 or 6583 is attached and B over C will be selected. If it is required to use the spectrum analyzer receiver, use :MEAS:MEAS:POW RX or :MEAS:MEAS:RAT RXC.

For a spectrum analyzer channel this command selects a live measurement using the receiver.

For a scalar channel this command generates an error. Use MEAS:MEAS:POW or MEAS:MEAS:RAT instead.

Example:
```
:MEAS:MEAS:LIVE
```

:MEASurement :MEASure :MEMory [:BINary] [:ONLY]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Display a memory. The memory store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

The settings stored with the memory are not recalled and therefore the memory is displayed using the current channel settings.

Example: `:MEAS:MEAS:MEM "FILTER"`

Display trace memory FILTER.

:MEASurement :MEASure :MEMory [:BINary] :SETTings

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Display a memory. The memory store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

The settings stored with the memory are recalled and therefore the memory is displayed exactly as it was stored. In some cases this may invalidate calibrations and prevent live measurements from updating.

Example: `:MEAS:MEAS:MEM:SETT "FILTER"`

Display trace memory FILTER.

:MEASurement
:MEASure
:MEMory
:ID

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name, id string

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

id string: string

Description: Set the user entered id string of the specified memory. The id will be applied to both a binary and a CSV memory of the specified name.

Example: `:MEAS:MEAS:MEM:ID "JOHN01", "Scalar 2-5GHz"`

:MEASurement
:MEASure
:MEMory
:ID?

Parameters: none

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the user entered id string of the memory associated with the active trace. The id string gives additional details of the memory store over what can be determined from the file name and extension alone.

Example: `:MEAS:MEAS:MEM:ID?`

:MEASurement :MEASure :POWer

Parameters: <CPD>

input id

Valid values: input id: [A | B | C | RX | GDELay]. Values other than those stated are rejected and an error generated.

Description: For a scalar channel, this sets up an absolute power measurement from the specified input which can be one of the three scalar inputs, the spectrum analyzer receiver or group delay (if this option is fitted).

For a fault location channel this command may only be used to select the spectrum analyzer receiver (i.e. :MEAS:MEAS:POW RX). This command is only valid if the active measurement is measurement 1. See :MEAS:MEAS:LIVE for selecting scalar inputs. See :MEAS:MEAS:RAT for selecting RX/C.

For a spectrum analyzer channel, this command gives an error. Use MEAS:MEAS:LIVE instead.

Example: `:MEAS:MEAS:POW A`

Measure absolute power from scalar input A.

:MEASurement :MEASure :RATio

Parameters: <CPD>

ratio

Valid values: ratio: [AC | BC | RXC]. Values other than those stated are rejected and an error generated.

Description: Set up a ratio measurement on the active measurement. The denominator is always C.

For scalar channels, this will set up a ratio measurement of A/C, B/C, or RX/C.

For fault location channels, this command will give an error. See :MEAS:MEAS:LIVE for selecting scalar inputs. See :MEAS:MEAS:POW for selecting RX.

For spectrum analyzer channels, this command gives an error since ratio measurements are not permitted. Use MEAS:MEAS:LIVE instead.

Example: `:MEAS:MEAS:RAT AC`

Measure the ratio A/C.

:MEASurement :MOPeration :SELect [:BINary]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Select a trace memory for use in a memory operation with the live measurement on the active measurement in the active channel. The memory store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.
The settings stored with the memory are not recalled and the memory is interpolated as necessary to fit with the live measurement.

An error will be generated if this command is used in a fault location or spectrum analyzer channel.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MEAS:MOP:SEL "TEST1"`

Select trace memory TEST1 *for use in a memory operation.*

:MEASurement :MOPeration [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable a ratio operation between a live measurement and a trace memory.
An error will be generated if this command is used in a fault location or spectrum analyzer channel.
An error will also be given if there is no memory associated with the active measurement when the memory operation is enabled.

Example: `:MEAS:MOP ON`

Enable the memory operation

:MEASurement :MOPeration [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether a memory operation is enabled.
An error will be generated if this command is used in a fault location or spectrum analyzer channel.

Example:
```
:MEAS:MOP?
```

:MEASurement :NMEas

Parameters: <NRf>

number of measurements

Valid values: number of measurements: integer. Valid values are 1 to 2. Values outside range are rejected and an error generated.

Description: Set the number of measurements displayed within the active channel.

Example:
```
:MEAS:NME 2
```

Display two measurements.

:MEASurement :NMEas?

Parameters: none

Response: <NR1>

number of measurements

Returned values: number of measurements: integer. Values are in the range 1 to 2.

Description: Determine the number of measurements displayed within the active channel.

Example:
```
:MEAS:NME?
```

:MEASurement :SAVE [:BINary]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Save the active measurement to a trace memory. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to floppy disk). A file extension should not be specified as this is fixed by the instrument.

Note that it is only possible to save live measurements to a trace memory.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A";:MEAS:SAVE "FILTER"`

Save trace data to a file named FILTER *on the floppy disk.*

:MEASurement :SAVE :TEXT

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Save the active measurement to a trace memory in Spreadsheet (Comma Separated Variable) format. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to floppy disk). A file extension should not be specified as this is fixed by the instrument.

Since it is not possible to display a CSV memory or use it in a memory operation, saving a CSV memory automatically saves a binary memory at the same time.

Note that it is only possible to save live measurements to a trace memory.

Use :SYST:ISET:DPO and :SYST:ISET:SEP to set the decimal point and field separator characters used when writing the values.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A";:MEAS:SAVE:TEXT "FILTER"`

Save trace data in spreadsheet format to a file named FILTER *on the floppy disk.*

:MEASurement :STATe

Parameters: <NRf>, <NRf>, <BOOLEAN PROGRAM DATA>

channel number, measurement number, state

Valid values: channel number: integer. Valid values are 1 to 2. Values outside range are rejected and an error generated.

measurement number: integer. Valid values are 1 to 2. Values outside range are rejected and an error generated.

Description: Set the displayed state of any of the four measurements that the instrument can display. This is an alternative to using the commands that only work on the active channel/active measurement.

If the command refers to a measurement in a non displayed channel then an error will be given.

Example: `:MEAS:STAT 2,1,ON`

Turn on measurement 1 in channel 2.

:MEASurement :STATe?

Parameters: <NRf>, <NRf>

channel number, measurement number

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if the specified measurement in the specified channel is displayed. If the command refers to a channel that is not displayed then an error will be given..

Example: `:MEAS:STAT? 2,2`

MMEMory subsystem

```
MMEMory
        ATTRibutes?
        CATalog
                COUNt?
                [FILes]?
                SIZes?
        CDIRectory\?
        COPY
        DELete
        ERASe
                [ALL]
        ID\?
        MDIRectory
        MSIS\?
        RDIRectory
        READ
                DBASe
                        CURRent?
                        [STORe]?
                HSETup?
                LIMit?
                MEMory
                        [BINary]?
                        TEXT?
                PCAL?
                SETTings?
                SOURce
                        FCALibration?
                        FM?
                        FSTandard?
                        PPOWer?
                        UPOWer?
        REName
        WRITe
                DBASe
                        [STORe]
                HSETup
                LIMit
                MEMory
                        [BINary]
                        TEXT
                PCAL
                SETTings
                SOURce
                        FCALibration
                        FM
                        FSTandard
                        PPOWer
                        UPOWer
```

:MMEMory

:ATTRibutes?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

attributes

Returned values: attributes: integer

Description: Return the attributes of the specified file. Both file name and extension must be given to ensure one and only one file will be accessed.

The returned value is encoded as follows:

Bit	Meaning
0	Store contains default data
1	Store is password protected
2	Read-only file
3	Archive
4	System
5	Hidden

The file will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:

```
:MMEM:MSIS "A"; CDIR "\TRACES"; ATTR? "MAXPK.TRC"
```

Determine attributes of MAXPK.TRC *in* \TRACES *on the floppy disk.*

:MMEMory
:CATalog?
:COUNt?

Parameters: <STRING PROGRAM DATA>

file type

Valid values: file type: string. Maximum length of 3 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

total files

Returned values: total files: integer

Description: Return a count of files of the type specified (or all, if the parameter is an asterisks "*"). To list directories, use "dir" as the file type.

The files will be accessed using the file type sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

The following extensions will be used to specify store types:

APP	Application Code Store
CDB	Fault Location cable stores
CSV	Trace memories (spreadsheet CSV format)
HCS	Hardcopy setting stores
LMT	Limit line stores
LNG	Foreign language support files
PCL	Scalar path calibration / Fault location calibration stores
PCS	User Power Calibration Stores
SET	Settings stores
TRC	Trace memories (binary format)

Example:
```
:MMEM:MSIS "A"; CAT:COUN? "TRC"
```

Determine how many trace memories are in the current directory on the floppy disk.

:MMEMory
:CATalog
[:FILes]?

Parameters: <STRING PROGRAM DATA>

file type

Valid values: file type: string. Maximum length of 3 characters excluding quotes. Excess characters will be ignored.

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first file name, ..., last file name

Returned values: first file name: string
...
last file name: string

Description: Return a list of files of the type specified (or all, if the parameter is an asterisks "*"). To list directories, use "dir" as the file type.

The files will be accessed using the file type sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

The following extensions will be used to specify store types:

APP	Application Code Store
CDB	Fault Location cable stores
CSV	Trace memories (spreadsheet CSV format)
HCS	Hardcopy setting stores
LMT	Limit line stores
LNG	Foreign language support files
PCL	Scalar path calibration / Fault location calibration stores
PCS	User Power Calibration Stores
SET	Settings stores
TRC	Trace memories (binary format)

Example: `:MMEM:MSIS "A"; CAT? "TRC"`

Get a catalogue of all the trace memories in the current directory on the floppy disk.

Example response: "HPF_3G.TRC", "RLOSS .TRC", "PEAK .TRC"

:MMEMory
:CATalog
:SIZes?

Parameters: <STRING PROGRAM DATA>

file type

Valid values: file type: string. Maximum length of 3 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>, ..., <NR1>

first file length, ..., last file length

Returned values: first file length: integer
...
last file length: integer

Description: Return a list of sizes of files of the type specified (or all, if the parameter is an asterisks "*"). To list directories, use "dir" as the file type.

The files will be accessed using the file type sent as part of this command together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

The following extensions will be used to specify store types:

APP	Application Code Store
CDB	Fault Location cable stores
CSV	Trace memories (spreadsheet CSV format)
HCS	Hardcopy setting stores
LMT	Limit line stores
LNG	Foreign language support files
PCL	Scalar path calibration / Fault location calibration stores
PCS	User Power Calibration Stores
SET	Settings stores
TRC	Trace memories (binary format)

Example:
```
:MMEM:MSIS "A"; CAT:COUN? "TRC"; SIZ? "TRC"
```

Get a catalogue of all the sizes of trace memories in the current directory on the floppy disk.

Example response: 3; 9754, 9754, 9754

:MMEMory :CDIRectory

Parameters: <STRING PROGRAM DATA>

directory path

Valid values: directory path: string. Maximum length of 144 characters excluding quotes. Excess characters will be ignored.

Description: Change the current directory. This only applies to the floppy disk. A null directory path will generate an error.

Example:
```
:MMEM:CDIR "\my_mems\scalar"
```

Set the current directory to be \my_mems\scalar.

:MMEMory :CDIRectory?

Parameters: none

Response: <STRING RESPONSE DATA>

directory path

Returned values: directory path: string

Description: Determine the current directory. This will always be returned as a full path name from root.

Example:
```
:MMEM:CDIR?
```

:MMEMory :COPY

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name source, file name destination

Valid values: file name source: string. Maximum length of 144 characters excluding quotes. Excess characters will be ignored.
file name destination: string. Maximum length of 144 characters excluding quotes. Excess characters will be ignored.

Description: Copy a file to another file. The file may be copied to and from the same device (floppy only) or across devices.

The file name must be a full pathname (except when accessing internal storage) and include the msus to specify one and only one file. Not all files can be accessed on the internal storage - only 'standard' files, e.g. trace memories, path cals, etc, can be copied. A path name must not be given for internal storage.

Use C: or A: to specify internal memory or floppy disk.

Example:
```
:MMEM:COPY "A:\MY_MEMS\SCALAR\FLAT_MEM.TRC",
"C:FLAT_MEM.TRC"
```

Copy trace memory from floppy disk to internal memory.

:MMEMory
:DELete

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

Description: Delete the specified file in the current directory. File name and extension must be given.

The file will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A"; CDIR "\TRACES"; DEL "MAXPK.TRC"`

Delete MAXPK.TRC *in* \TRACES *on the floppy disk.*

:MMEMory
:ERASe
[:ALL]

Parameters: none

Description: Erase all stores. The following internal stores are cleared:

All measurement stores
All settings stores

Example: `:MMEM:ERAS`

Erase most of the stores in the instrument.

:MMEMory

:ID

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name, user id

Valid values: file name: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

user id: string

Description: Set the id string of the specified file in the current directory. The id string gives additional details of the file over what can be determined from the file name and extension alone. For most files the id string is user entered - allowing a more meaningful description than that allowed by the DOS filename.

Both file name and extension must be given to ensure one and only one file will be accessed.

The file will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:
```
:MMEM:MSIS "A"; CDIR "\TRACES"; ID "MAXPK.TRC", "Johns"
```

Set the user entered id string from MAXPK.TRC *in* \TRACES *on the floppy disk.*

:MMEMory
:ID?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the id string of the specified file in the current directory. The id string gives additional details of the file over what can be determined from the file name and extension alone. For most files the id string is user entered - allowing a more meaningful description than that allowed by the DOS filename. For fixed files, the id string will give information appropriate to the file type, for example with hardcopy devices (.DRV files) the id string will give the name of the printer(s) that the driver supports.

Both file name and extension must be given to ensure one and only one file will be accessed.

The file will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk).

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:

```
:MMEM:MSIS "A"; CDIR "\TRACES"; ID? "MAXPK.TRC"
```

Read the user entered id string from MAXPK.TRC *in* \TRACES *on the floppy disk.*

:MMEMory :MDIRectory

Parameters: <STRING PROGRAM DATA>

directory name

Valid values: directory name: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

Description: Create a new directory as a subdirectory of the current directory. This only applies to the floppy disk.

Use :MMEM:CDIR to select the directory to create the new directory in.

Example: `:MMEM:MDIR "scalar"`

Create a new directory called scalar *as a subdirectory of the current directory.*

:MMEMory :MSIS

Parameters: <STRING PROGRAM DATA>

storage device

Valid values: storage device: string. Maximum length of 1 character excluding quotes. Excess characters will be ignored.

Description: Change the currently selected mass storage device.

The storage device is defined as:

A	The floppy disk.
C	The internal memory.

Example: `:MMEM:MSIS "A"`

Set the selected mass storage device to be floppy disk.

:MMEMory :MSIS?

Parameters: none

Response: <STRING RESPONSE DATA>

storage device

Returned values: storage device: string. Maximum length of 1 character excluding quotes.

Description: Determine the currently selected mass storage device.

The storage device is defined as:

A	The floppy disk.
C	The internal memory.

Example: `:MMEM:MSIS?`

:MMEMory
:RDIRectory

Parameters: <STRING PROGRAM DATA>

directory name

Valid values: directory name: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

Description: Delete the specified directory, which must be a subdirectory of the current directory and must be empty. This only applies to the floppy disk.

Use :MMEM:CDIR to select the parent directory of the directory you wish to remove.

Example:
```
:MMEM:RDIR "scalar"
```
Delete a directory called scalar *which is a subdirectory of the current directory.*

:MMEMory
:READ
:DBASe
:CURRent?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

current cable in use data

Description: Read cable data currently in use on the active channel and output over remote interface.

Example:
```
:MMEM:READ:DBAS:CURR?
```
Read cable data currently in use.

:MMEMory :READ :DBASe [:STORe]?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

fault location cable store contents

Description: Read cable data from a fault location cable store and output over remote interface. The store will be accessed using the file name sent as part of this command. These stores cannot be stored on the floppy disk. A file extension should not be specified as this is fixed by the instrument.

Example: `:MMEM:READ:DBAS? "AND_LDF2"`

Read cable data from store AND_LDF2 *in internal memory.*

:MMEMory :READ :HSETup?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

hardcopy setup store contents

Description: Read hardcopy setup data from a hardcopy setup store and output over remote interface. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A"; READ:HSET? "LAYOUT"`

Read hardcopy setup data from store LAYOUT *on the floppy disk.*

:MMEMory
:READ
:LIMit?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

limit store contents

Description: Read limit data from a limit specification store and output over remote interface. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A"; READ:LIM? "RLOSS"`

Read limit specification data from store RLOSS *on the floppy disk.*

:MMEMory
:READ
:MEMory
[:BINary]?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

memory contents

Description: Read trace memory data from a trace memory store and output over remote interface. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A"; READ:MEM? "RLOSS"`

Read trace memory data from store RLOSS *on the floppy disk.*

:MMEMory
:READ
:MEMory
:TEXT?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

memory contents

Description: Read trace memory data from a CSV trace memory store and output over remote interface. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:
```
:MMEM:MSIS "A"; READ:MEM:TEXT? "RLOSS"
```

Read CSV format trace memory data from store RLOSS *on the floppy disk.*

:MMEMory
:READ
:PCAL?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

path calibration store contents

Description: Read path calibration data from a path calibration store and output over remote interface. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error. When using path calibrations/fault location calibrations on a floppy disk, any filename may be used.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "C"; READ:PCAL? "PCL3"`

Read path calibration data from internal store PCL3.

:MMEMory
:READ
:SETTings?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

settings store contents

Description: Read instrument settings data from a settings store and output over remote interface. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A"; READ:SETT? "RLOSS"`

Read settings data from store RLOSS *on the floppy disk.*

:MMEMory
:READ
:SOURce
:FCALibration?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

store contents

Description: Read the source frequency calibration store data.

Example: `:MMEM:READ:SOUR:FCAL?`

:MMEMory :READ :SOURce :FM?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

store contents

Description: Read the source fm calibration store data. This command will generate an error if the FM option is not present.

Example: `:MMEM:READ:SOUR:FM?`

:MMEMory :READ :SOURce :FSTandard?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

store contents

Description: Read the source frequency standard calibration store data.

Example: `:MMEM:READ:SOUR:FST?`

:MMEMory :READ :SOURce :PPOWer?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

Description: Read the source primary power calibration store data.

Example: `:MMEM:READ:SOUR:PPOW?`

:MMEMory :READ :SOURce :UPOWer?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

store contents

Description: Read data from user source power calibration store.

Example: `:MMEM:READ:SOUR:UPOW? "USR2"`

Read power calibration data from user source power calibration store USR2.

:MMEMory :REName

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name old, file name new

Valid values: file name old: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.
file name new: string. Maximum length of 12 characters excluding quotes. Excess characters will be ignored.

Description: Rename a file. The full filename including the extension must be specified for the old name and the new name. It is not recommended that the extension is changed since the instrument uses the extension to determine the type of file being accessed.

The files will be accessed using the file names sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if the file to be renamed is on the floppy disk).

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:REN "FLAT_MEM.TRC", "NEW_MEM.TRC"`

Rename a trace memory file from FLAT_MEM *to* NEW_MEM.

:MMEMory :WRITe :DBASe [:STORe]

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, cable data

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write cable data to a fault location cable store. The store will be accessed using the file name sent as part of this command. These stores cannot be stored on the floppy disk. A file extension should not be specified as this is fixed by the instrument.

Example: `:MMEM:WRIT:DBAS "FLEXY",#...etc`

Write cable data to store FLEXY *in the instrument. (Only the first character of the cable data is shown).*

:MMEMory :WRITe :HSETup

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, hardcopy setup store data

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write hardcopy setup data to a hardcopy setup store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "C"; WRIT:LIM "FOR_HP",#...etc`

Write hardcopy setup data to store FOR_HP *in the instrument. (Only the first character of the hardcopy setup data is shown).*

:MMEMory
:WRITe
:LIMit

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, limit data

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write limit data to a limit specification store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:
```
:MMEM:MSIS "C"; WRIT:LIM "INS_LOSS",#...etc
```

Write limit specification data to store INS_LOSS *in the instrument. (Only the first character of the limit data is shown).*

:MMEMory
:WRITe
:MEMory
[:BINary]

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, memory contents

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write trace memory data to a trace memory store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:
```
:MMEM:MSIS "A"; WRIT:MEM "RLOSS",#...etc
```

Write trace memory data to store RLOSS *on the floppy disk. (Only the first character of the memory data is shown).*

:MMEMory :WRITe :MEMory :TEXT

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, memory contents

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write trace memory data to a CSV trace memory store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "A"; WRIT:MEM:TEXT "RLOSS",#...etc`

Write CSV trace memory data to store RLOSS *on the floppy disk. (Only the first character of the memory data is shown).*

:MMEMory :WRITe :PCAL

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, path calibration data

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write path calibration data to a path calibration store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error. When using path calibrations/fault location calibrations on a floppy disk, any filename may be used.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example: `:MMEM:MSIS "C"; WRIT:PCAL "PCL2",#...etc`

Write path calibration data to store PCL2 *in the instrument. (Only the first character of the path calibration data is shown).*

:MMEMory
:WRITe
:SETTings

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, settings store contents

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Write instrument settings data to a settings store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if saving to the floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select internal memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

Example:
```
:MMEM:MSIS "A"; WRIT:SETT "RLOSS",#...etc
```

Write instrument settings data to store RLOSS *on the floppy disk. (Only the first character of the memory data is shown).*

:MMEMory
:WRITe
:SOURce
:FCALibration

Parameters: <ARBITRARY BLOCK PROGRAM DATA>

store contents

Description: Send data to source frequency calibration store.

Example:
```
:MMEM:WRIT:SOUR:FCAL #...etc
```

Send source frequency calibration data to source frequency store. (Only first byte of data shown).

:MMEMory
:WRITe
:SOURce
:FM

Parameters: <ARBITRARY BLOCK PROGRAM DATA>

store contents

Description: Send data to source FM calibration store. This command will give an error if the FM option is not fitted.

Example: `:MMEM:WRIT:SOUR:FM #...etc`

Send source FM calibration data to source fm store. (Only first byte of data shown).

:MMEMory
:WRITe
:SOURce
:FSTandard

Parameters: <ARBITRARY BLOCK PROGRAM DATA>

store contents

Description: Send data to source frequency standard calibration store.

Example: `:MMEM:WRIT:SOUR:FST #...etc`

Send source frequency calibration data to source frequency store. (Only first byte of data shown).

:MMEMory
:WRITe
:SOURce
:PPOWer

Parameters: <ARBITRARY BLOCK PROGRAM DATA>

store contents

Description: Write the source primary power calibration store data.

Example: `:MMEM:WRIT:SOUR:PPOW #...etc`

:MMEMory
:WRITe
:SOURce
:UPOWer

Parameters: <STRING PROGRAM DATA>,<ARBITRARY BLOCK PROGRAM DATA>

file name, store contents

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Send data to user source power calibration store.

Example: `:MMEM:WRIT:SOUR:UPOW "CAL1", #...etc`

Send source power calibration data to user source power calibration store CAL1. *(Only first byte of data shown).*

SANalyzer subsystem

```
SANalyzer
    ATTenuation
        AUTO\?
        VALue\?
    ATUNing
    CALibrate
        AMPLitude
        DETector
        ESOurce\?
            CURRent?
            DISK
                INSTall
                LIST?
                NUMBer?
            FREQuency
                DEFault
                VALue\?
            LIST?
            NUMBer?
            SELect
            [STATe]\?
        LO
        PSELector
        RBFilter
    COUNter
        RESolution\?
        [STATe]\?
    DEModulation
        AUDio\?
        DISPlay\?
        FREQuency\?
        MODE\?
        TIMebase\?
    EMIXer
        CLOSs\?
        RANGe\?
        [STATe]\?
    NORMalise
        OFF
        [PERForm]
    PEAK
        RESTart
        [STATe]\?
    RBANdwidth
        AUTO\?
        VALue\?
```

RX
    CENTer\?
    CENTre\?
    SPAN\?
    STARt\?
    STOP\?
SWEep
    AUTO\?
    VALue\?
TGENerator
    [MODE]\?
    OFFSet\?
    ORDer\?
    SCALing\?
TRACking\?
VBANdwidth
    AUTO\?
    VALue\?

:SANalyzer
:ATTenuation
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set whether the user or the instrument sets the input attenuation for the active channel. This command is only valid for spectrum analyzer channels.
If the instrument is automatically setting the input attenuation then changing the reference value may alter the input attenuation.

Example: `:SAN:ATT:AUTO ON`

The instrument sets the input attenuation on the active channel.

:SANalyzer
:ATTenuation
:AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the user or the instrument sets the input attenuation for the active channel. This command is only valid for spectrum analyzer channels.

Example: `:SAN:ATT:AUTO?`

:SANalyzer
:ATTenuation
:VALue

Parameters: <NUMERIC VALUE>

input attenuation

Valid values: input attenuation: real

Suffix: input attenuation: A suffix of dB is accepted for attenuation. If no suffix is entered then the default suffix of dB is assumed.

Description: Set the input attenuation for the active channel. This command is only valid for spectrum analyzer channels.
This will automatically switch into 'user set attenuation' mode (i.e. an implied :SAN:ATT:AUTO OFF is sent first).
Since the input attenuation is settable in discrete steps, the entered value will be rounded to the nearest valid value.
See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:ATT:VAL 3`

Set an input attenuation of 3 dB on the active channel.

:SANalyzer :ATTenuation :VALue?

Parameters: none

Response: <NR2>

input attenuation

Returned values: input attenuation: real

Description: Determine the input attenuation for the active channel. This command is only valid for spectrum analyzer channels.
Note that this command will return the current value of attenuation regardless of whether it was set automatically or by the user.

Example: `:SAN:ATT:VAL?`

:SANalyzer :ATUNing

Parameters: none

Description: Perform automatic tuning, which sets up a spectrum analyzer measurement automatically. The display is set up to show the largest peak.

Example: `:SAN:ATUN`

:SANalyzer :CALibrate :AMPLitude

Parameters: none

Description: Perform a spectrum analyzer amplitude calibration. This removes frequency response errors of all the front end components.

Use SAN:CAL:ESO to select either the internal or the external source to perform the calibration with. If the external source is selected then a 6203 is required for variants that allow the spectrum analyser to operate up to 21 GHz and a 6204 is required for those variants operating up to 30 GHz.

Example: `:SAN:CAL:AMPL`

:SANalyzer :CALibrate :DETector

Parameters: none

Description: Perform a spectrum analyzer log amplifier/detector calibration.

Example: `:SAN:CAL:DET`

:SANalyzer
:CALibrate
:ESOurce
:CURRent?

Parameters: none

Response: <STRING RESPONSE DATA>

external source in use

Returned values: external source in use: string

Description: Determine which external source is in use for the preselector and amplitude calibrations. This string will be the same as that used to select the external source using :SAN:CAL:ESO:SEL.

Example:
```
:SAN:CAL:ESO:CURR?
```

:SANalyzer
:CALibrate
:ESOurce
:DISK
:INSTall

Parameters: <STRING PROGRAM DATA>

external source

Valid values: external source: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Install a new external source driver into the instrument. The external source string must be one of the strings read using :SAN:CAL:ESO:DISK:LIST? or an error will be generated. Once the new external source has been installed into the instrument it can be selected using :SAN:CAL:ESO:SEL.

Example:
```
:SAN:CAL:ESO:DISK:INST "6204"
```

Install 6204 driver into the instrument.

:SANalyzer
:CALibrate
:ESOurce
:DISK
:LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first external source, ..., last external source

Returned values: first external source: string. Maximum length of 30 characters excluding quotes.
...
last external source: string. Maximum length of 30 characters excluding quotes.

Description: List all external source drivers on the floppy disk that are available to be installed into the instrument. The disk containing the external source(s) to be installed must be present. To install one of the external sources listed, use :SAN:CAL:ESO:DISK:INST using one of the strings returned by this command.

Example: `:SAN:CAL:ESO:DISK:LIST?`

List all the external source drivers on the floppy disk.

:SANalyzer
:CALibrate
:ESOurce
:DISK
:NUMBer?

Parameters: none

Response: <NR1>

number of external sources

Returned values: number of external sources: integer

Description: This command is to be used in conjunction with :SAN:CAL:ESO:DISK:LIST. This command determines how many external source drivers will be returned by the :SAN:CAL:ESO:DISK:LIST? command. This can be useful in reserving space for the :SAN:CAL:ESO:DISK:LIST? response.

Example: `:SAN:CAL:ESO:DISK:NUMB?`

Determine how many external source drivers are on the floppy disk

:SANalyzer
:CALibrate
:ESOurce
:FREQuency
:DEFault

Parameters: none

Description: Set the maximum frequency to be used for the preselector or amplitude calibrations when using an external source to the maximum that the external source can handle.

This command is only valid if external source control has been turned on, by using SAN:CAL:ESO ON.

Example: `:SAN:CAL:ESO:FREQ:DEF`

Set the calibration max frequency to be the maximum of the external source.

:SANalyzer
:CALibrate
:ESOurce
:FREQuency
:VALue

Parameters: <NUMERIC VALUE>

max source frequency

Valid values: max source frequency: real

Suffix: max source frequency: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the maximum frequency to be used for the preselector or amplitude calibrations when using an external source. This value should be less than the maximum that the external source can handle.

Example: `:SAN:CAL:ESO:FREQ:VAL 22GHz`

Set the calibration max frequency to be 22 GHz.

:SANalyzer
:CALibrate
:ESOurce
:FREQuency
:VALue?

Parameters: none

Response: <NR2>

max source frequency

Returned values: max source frequency: real

Description: Determine the maximum frequency to be used for preselector and amplitude calibrations when using an external source.

Example:
```
:SAN:CAL:ESO:FREQ:VAL?
```

:SANalyzer
:CALibrate
:ESOurce
:LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first external source, ..., last external source

Returned values: first external source: string. Maximum length of 30 characters excluding quotes.
...
last external source: string. Maximum length of 30 characters excluding quotes.

Description: List all external source drivers available in the instrument. An external source may then be selected using :SAN:CAL:ESO:SEL with one of the strings returned by this command.

Example:
```
:SAN:CAL:ESO:LIST?
```

List all the external source drivers available.

:SANalyzer
:CALibrate
:ESOurce
:NUMBer?

Parameters: none

Response: <NR1>

number of external sources

Returned values: number of external sources: integer

Description: This command is to be used in conjunction with :SAN:CAL:ESO:LIST. This command determines how many external source drivers will be returned by the :SAN:CAL:ESO:LIST? command. This can be useful in reserving space for the :SAN:CAL:ESO:LIST? response.

Example:
```
:SAN:CAL:ESO:NUMB?
```
Determine how many external source drivers are available in the instrument.

:SANalyzer
:CALibrate
:ESOurce
:SELect

Parameters: <STRING PROGRAM DATA>

external source

Valid values: external source: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Select a new external source driver. The external source string must be one of the strings read using :SAN:CAL:ESO:LIST? or an error will be generated.

Example:
```
:SAN:CAL:ESO:SEL "6204"
```
Select the external source to be a 6204.

:SANalyzer
:CALibrate
:ESOurce
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

Use External Source

Description: This selects either the internal or the external source when performing :SAN:CAL:AMPL and SAN:CAL:PSEL.

Note that passing control is not supported when using an external source to perform either of these calibrations, thus the commands relating to external source control for these two calibrations are only relevant when controlling the instrument from rs232.

Example: `:SAN:CAL:ESO ON; AMPL`

Perform an amplitude cal using an external source.

:SANalyzer
:CALibrate
:ESOurce
[:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

Use External Source

Description: Determine whether the internal or external source will be used for spectrum analyser amplitude and preselector (yig filter) cals.

Example: `:SAN:CAL:ESO?`

:SANalyzer
:CALibrate
:LO

Parameters: none

Description: Perform a spectrum analyzer 1st LO frequency calibration.

Example: `:SAN:CAL:LO`

:SANalyzer :CALibrate :PSELector

Parameters: none

Description: Perform a spectrum analyzer YIG preselector filter calibration.

Use SAN:CAL:ESO to select either the internal or the external source to perform the calibration with. If the external source is selected then a 6203 is required for variants that allow the spectrum analyser to operate up to 21 GHz and a 6204 is required for those variants operating up to 30 GHz.

Example:
```
:SAN:CAL:ESO OFF; PRES
```

:SANalyzer :CALibrate :RBFilter

Parameters: none

Description: Perform a spectrum analyzer resolution bandwidth filter calibration.

Example:
```
:SAN:CAL:RBF
```

:SANalyzer :COUNter :RESolution

Parameters: <NUMERIC VALUE>

counter resolution

Valid values: counter resolution: real

Suffix: counter resolution: The following suffixes are accepted for frequency: Hz, kHz and MHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the counter resolution. The value is in Hz. The value will be rounded to the nearest valid value. See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:SAN:COUN:RES 1E3
```

Set counter resolution to 1KHz.

:SANalyzer :COUNter :RESolution?

Parameters: none

Response: <NR2>

counter resolution

Returned values: counter resolution: real

Description: Determine the counter resolution. The value is in Hz.

Example: `:SAN:COUN:RES?`

:SANalyzer :COUNter [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable or disable counter facility. This command will produce an error if the active channel is not a spectrum analyzer channel. When the counter is enabled, the active marker frequency readout on the active channel (if it is a spectrum analyzer channel) will be updated with the counter reading once per measurement update.

Example: `:SAN:COUN ON`

Turn on counter function.

:SANalyzer :COUNter [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether counter is on or off.

Example: `:SAN:COUN?`

:SANalyzer
:DEModulation
:AUDio

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable speaker output of demodulated signal. The volume of the speaker is set using a hard control on the front panel of the instrument. The speaker will produce the demodulated signal independently of whether the screen is showing the spectrum analyzer waveform in demodulated mode or not.

Use :SAN:DEM:MODE to set the demodulation mode.

If there are no displayed spectrum analyzer channels then the speaker will not be driven even if it is turned on.
If there is one displayed spectrum analyzer channel then the speaker will be driven using the active marker frequency of that channel.
If there are two displayed spectrum analyzer channels then the active channel's active marker frequency will be used to drive the speaker.

Example: `:SAN:DEM:AUD ON`

Turn on loudspeaker and play demodulated signal.

:SANalyzer
:DEModulation
:AUDio?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if loudspeaker is being used to produce an audible demodulated signal.

Example: `:SAN:DEM:AUD?`

:SANalyzer
:DEModulation
:DISPlay

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable demodulated display. This applies on a per channel basis.

Use :SAN:DEM:MODE to set the demodulation mode. Use :SAN:DEM:TIM to set the timebase for the demodulated display. The demodulation will be performed at the active marker frequency.

Example: `:SAN:DEM:DISP ON`

Show 'oscilloscope' format - a demodulated waveform of amplitude vs time.

:SANalyzer :DEModulation :DISPlay?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if channel is displaying a demodulated waveform or not.

Example: `:SAN:DEM:DISP?`

:SANalyzer :DEModulation :FREQuency

Parameters: <NUMERIC VALUE>

frequency value

Valid values: frequency value: real

Suffix: frequency value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a spectrum analyzer channel, this command will set the demodulation frequency. This command is only valid when demodulation is enabled.

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:DEM:FREQ 5E9`

Set the demodulation frequency to 5 GHz.

:SANalyzer :DEModulation :FREQuency?

Parameters: none

Response: <NR2>

frequency value

Returned values: frequency value: real

Description: Determine the demodulation frequency value for the active channel. An error will be generated if the channel is not a spectrum analyzer channel in demodulation mode.

Example: `:SAN:DEM:FREQ?`

:SANalyzer :DEModulation :TIMebase

Parameters: <NUMERIC VALUE>

timebase

Valid values: timebase: real

Suffix: timebase: A suffix of either ms or s is accepted for time. If no suffix is entered then the default suffix of s is assumed.

Description: Set the timebase for the demodulated display. The value is in seconds per division. This value will only be used when the spectrum analyzer channel has been set into a demodulated display using :SAN:DEM:DISP ON. See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:DEM:TIM 0.01`

Set a timebase to 10 ms per div.

:SANalyzer :DEModulation :TIMebase?

Parameters: none

Response: <NR2>

timebase

Returned values: timebase: real

Description: Determine the timebase used for the demodulated display.

Example: `:SAN:DEM:TIM?`

:SANalyzer :EMIXer :CLOSs

Parameters: <NRf>

conversion loss

Valid values: conversion loss: real

Description: Set the conversion loss for the external mixer (in units of dB). This command will generate an error if the active channel is not a spectrum analyzer channel.

Example: `:SAN:EMIX:CLOS 15`

Set external mixer conversion loss to 15dB.

:SANalyzer :EMIXer :CLOSs?

Parameters: none

Response: <NR2>

conversion loss

Returned values: conversion loss: real

Description: Determine the conversion loss of the external mixer (in dB). This command will generate an error if the active channel is not a spectrum analyzer channel.

Example:
```
:SAN:EMIX:CLOS?
```

:SANalyzer :EMIXer :RANGe

Parameters: <NUMERIC VALUE>, <NUMERIC VALUE>

lower freq limit, upper freq limit

Valid values: lower freq limit: real

upper freq limit: real

Suffix: lower freq limit: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

upper freq limit: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the external mixer operating range. See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:SAN:EMIX:RANG 30E6, 1E9
```

Set the external mixer range to be 30 MHz to 1 GHz.

:SANalyzer :EMIXer :RANGe?

Parameters: none

Response: <NR2>, <NR2>

lower freq limit, upper freq limit

Returned values: lower freq limit: real

upper freq limit: real

Description: Determine the range of the external mixer. The values returned are in units of Hz.

Example:
```
:SAN:EMIX:RANG?
```

:SANalyzer :EMIXer [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable or disable external mixer. This command will produce an error if the active channel is not a spectrum analyzer channel.

Example: `:SAN:EMIX ON`

Enable external mixer.

:SANalyzer :EMIXer [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether an external mixer is being used.

Example: `:SAN:EMIX?`

:SANalyzer :NORMalise :OFF

Parameters: none

Description: Turn off spectrum analyzer normalisation. When normalisation is turned off the reference level is set back to what it was originally (i.e. the value before the normalisation was done).

Example: `:SAN:NORM:OFF`

:SANalyzer
:NORMalise
[:PERForm]

Parameters: none

Description: Perform (and apply) spectrum analyzer normalisation.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the normalisation has completed. If the normalisation is being performed in triggered mode then it is necessary to read bit 3 or bit 5 of the operation status register to determine when the normalisation has completed. A normalisation requires one sweep only.

Any command received whilst the normalisation is in progress that cannot be actioned because it would affect the normalisation will generate an error.

Example: `:SAN:NORM`

:SANalyzer
:PEAK
:RESTart

Parameters: none

Description: Restart the peak hold function. This will clear the held values, thus allowing the peaks to build up from scratch.

Example: `:SAN:PEAK:REST`

:SANalyzer
:PEAK
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable or disable peak hold for the active measurement. When enabled, the instrument shall store and display a new value at each frequency of a sweep only if it exceeds the previously stored value at that frequency.

Example: `:SAN:PEAK ON`

Turn on peak hold function.

:SANalyzer :PEAK [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether peak hold is on or off.

Example:
```
:SAN:PEAK?
```

:SANalyzer :RBANdwidth :AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set whether the user or the instrument sets the resolution bandwidth for spectrum analyzer channels. If the instrument is automatically setting the resolution bandwidth then changing start/stop/centre/span frequencies may alter the resolution bandwidth.

Example:
```
:SAN:RBAN:AUTO ON
```

The instrument sets the resolution bandwidth.

:SANalyzer :RBANdwidth :AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the user or the instrument sets the resolution bandwidth for spectrum analyzer channels.

Example:
```
:SAN:RBAN:AUTO?
```

:SANalyzer
:RBANdwidth
:VALue

Parameters: <NUMERIC VALUE>

resolution bandwidth

Valid values: resolution bandwidth: real

Suffix: resolution bandwidth: The following suffixes are accepted for frequency: Hz, kHz and MHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the resolution bandwidth for spectrum analyzer channels. This will automatically switch into 'user set resolution bandwidth' mode (i.e. an implied :SAN:RBAN:AUTO OFF is sent first). Since resolution bandwidth is settable in discrete steps, the entered value will be rounded to the nearest valid value. See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:RBAN:VAL 3E6`

Set a resolution bandwidth of 3 MHz.

:SANalyzer
:RBANdwidth
:VALue?

Parameters: none

Response: <NR2>

resolution bandwidth

Returned values: resolution bandwidth: real

Description: Determine the resolution bandwidth (in Hz) for the active spectrum analyzer channel. Note that this command will return the current value of resolution bandwidth regardless of whether it was set automatically or by the user.

Example: `:SAN:RBAN:VAL?`

:SANalyzer
:RX
:CENTer

Parameters: <NUMERIC VALUE>

frequency centre value

Valid values: frequency centre value: real

Suffix: frequency centre value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a spectrum analyzer channel, this command will set the receiver centre frequency.

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:RX:CENT 15E9`

Set the receiver centre frequency to 15 GHz.

:SANalyzer
:RX
:CENTer?

Parameters: none

Response: <NR2>

frequency centre value

Returned values: frequency centre value: real

Description: Determine the receiver frequency centre value for the active channel. An error will be generated if the channel is not a spectrum analyzer channel.

Example: `:SAN:RX:CENT?`

:SANalyzer
:RX
:CENTre

Parameters: <NUMERIC VALUE>

frequency centre value

Valid values: frequency centre value: real

Suffix: frequency centre value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a spectrum analyzer channel, this command will set the receiver centre frequency.

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:RX:CENT 15E9`

Set the receiver centre frequency to 15 GHz.

:SANalyzer
:RX
:CENTre?

Parameters: none

Response: <NR2>

frequency centre value

Returned values: frequency centre value: real

Description: Determine the receiver frequency centre value for the active channel. An error will be generated if the channel is not a spectrum analyzer channel.

Example: `:SAN:RX:CENT?`

:SANalyzer
:RX
:SPAN

Parameters: <NUMERIC VALUE>

frequency span value

Valid values: frequency span value: real

Suffix: frequency span value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a spectrum analyzer channel, this command will set the receiver span.

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:RX:SPAN 10E9`

Set the receiver frequency span to 10 GHz.

:SANalyzer
:RX
:SPAN?

Parameters: none

Response: <NR2>

span value

Returned values: span value: real

Description: Determine the receiver frequency span. An error will be generated if the channel is not a spectrum analyzer channel.

Example: `:SAN:RX:SPAN?`

:SANalyzer
:RX
:STARt

Parameters: <NUMERIC VALUE>

frequency start value

Valid values: frequency start value: real

Suffix: frequency start value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a spectrum analyzer channel, this command will set the receiver start frequency.

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:RX:STAR 10.0E9`

Set the receiver start frequency to 10 GHz.

:SANalyzer
:RX
:STARt?

Parameters: none

Response: <NR2>

frequency start value

Returned values: frequency start value: real

Description: Determine the receiver frequency start value for the active channel. An error will be generated if the channel is not a spectrum analyzer channel.

Example: `:SAN:RX:STAR?`

:SANalyzer
:RX
:STOP

Parameters: <NUMERIC VALUE>

frequency stop value

Valid values: frequency stop value: real

Suffix: frequency stop value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a spectrum analyzer channel, this command will set the receiver stop frequency.

For any other channel, an error will be generated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:RX:STOP 16.5E9`

Set the receiver stop frequency to 16.5 GHz.

:SANalyzer
:RX
:STOP?

Parameters: none

Response: <NR2>

frequency stop value

Returned values: frequency stop value: real

Description: Determine the receiver frequency stop value for the active channel. An error will be generated if the channel is not a spectrum analyzer channel.

Example: `:SAN:RX:STOP?`

:SANalyzer
:SWEep
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable automatic setting of sweep time for the active channel. This command applies to spectrum analyzer channels only.
If channel coupling is on, the non active channel is also affected.

Example: `:SAN:SWE:AUTO OFF`

Enable spectrum analyzer sweep time to be set by the user.

:SANalyzer :SWEep :AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether spectrum analyzer sweep time is set automatically for the active channel.

Example:
```
:SAN:SWE:AUTO?
```

:SANalyzer :SWEep :VALue

Parameters: <NUMERIC VALUE>

sweep time

Valid values: sweep time: real

Suffix: sweep time: A suffix of either ms or s is accepted for time. If no suffix is entered then the default suffix of s is assumed.

Description: Set the sweep time for the active channel. This command applies to spectrum analyzer channels only. If channel coupling is on, the non active channel is also affected. Setting a user sweep time will disable the automatic sweep time function.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:SAN:SWE:VAL 0.5
```

Set a sweep time of 500 ms.

:SANalyzer :SWEep :VALue?

Parameters: none

Response: <NR2>

sweep time

Returned values: sweep time: real

Description: Determine the sweep time (in seconds) for the active channel. Note that this command will return the current value of sweep time regardless of whether it was set automatically or by the user.

Example:
```
:SAN:SWE:VAL?
```

:SANalyzer :TGENerator [:MODE]

Parameters: <CPD>

tracking generator mode

Valid values: tracking generator mode: [CW | TRACking | OTRacking]. Values other than those stated are rejected and an error generated.

Description: Set up the tracking generator facility on the active channel. This command will produce an error if the active channel is not a spectrum analyzer channel.

If the tracking generator is enabled (TRAC) then the source does a linear frequency sweep over the same range as the receiver.

If the tracking generator is enabled (OTR) then the source does a linear frequency sweep using a scale and offset from the receiver start and stop frequencies. See :SAN:TGEN:OFFS and :SAN:TGEN:SCAL.

If the tracking generator is disabled (CW), the source is set into a CW mode. The frequency can be set using :SOUR:FREQ:CW.

Example: `:SAN:TGEN TRAC`

Turn on tracking generator function to match the receiver frequency range.

:SANalyzer :TGENerator [:MODE]?

Parameters: none

Response: <CRD>

tracking generator mode

Returned values: tracking generator mode: [CW | TRAC | OTR]

Description: Determine the tracking generator mode for the active channel. This command will produce an error if the active channel is not a spectrum analyzer channel.

Example: `:SAN:TGEN?`

:SANalyzer :TGENerator :OFFSet

Parameters: <NUMERIC VALUE>

frequency offset

Valid values: frequency offset: real

Suffix: frequency offset: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the frequency offset from the receiver to the tracking generator for the active channel. This command will generate an error if the active channel is not a spectrum analyzer channel.
Note that this value will only be used if the source is set into offset tracking mode using :SAN:TGEN OTR.

Example: `:SAN:TGEN:OFFS -5.0E+9`

Offset tracking generator frequency by -5 GHz.

:SANalyzer :TGENerator :OFFSet?

Parameters: none

Response: <NR2>

frequency offset

Returned values: frequency offset: real

Description: Determine the frequency offset from the receiver to the tracking generator for the active channel. The offset value is in Hz. This command will generate an error if the active channel is not a spectrum analyzer channel.

Example: `:SAN:TGEN:OFFS?`

SANalyzer :TGENerator :ORDer

Parameters: <CPD>

scaling offset order

Valid values: scaling offset order: [SOFFset | OSCaling]. Values other than those stated are rejected and an error generated.

Description: Set up the order that scaling and offset are performed on the active channel. This command will produce an error if the active channel is not a spectrum analyzer.

This command only has any affect when the tracking generator is in OTR mode.

SOFF will perform scaling before offsetting and OSC will do the opposite.

Example: `:SAN:TGEN:ORD SOFF`

Perform scaling before offsetting.

:SANalyzer :TGENerator :ORDer?

Parameters: none

Response: <CRD>

scaling offset order

Returned values: scaling offset order: [SOFF | OSC]

Description: Determine the order of scaling/offset for the active channel. This command will produce an error if the active channel is not a spectrum analyzer channel.

Example: `:SAN:TGEN:ORD?`

:SANalyzer :TGENerator :SCALing

Parameters: <NRf>

frequency scale factor

Valid values: frequency scale factor: real

Description: Set the frequency scale factor from the receiver to the tracking generator for the active channel. This command will generate an error if the active channel is not a spectrum analyzer channel.
Note that this value will only be used if the source is set into offset tracking mode using :SAN:TGEN OTR.

Example: `:SAN:TGEN:SCAL 1.5`

Apply a frequency scale factor of 1.5.

:SANalyzer :TGENerator :SCALing?

Parameters: none

Response: <NR2>

frequency scale factor

Returned values: frequency scale factor: real

Description: Determine the frequency scale factor from the receiver to the tracking generator for the active channel. This command will generate an error if the active channel is not a spectrum analyzer channel.

Example:
```
:SAN:TGEN:SCAL?
```

:SANalyzer :TRACking

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: This sets signal tracking on or off. When enabled, the frequency range will be adjusted at the end of each sweep in order to place the largest detected signal at the centre of the display.

Example:
```
:SAN:TRAC ON
```

Enable signal tracking.

:SANalyzer :TRACking?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine if signal tracking is enabled.

Example:
```
:SAN:TRAC?
```

:SANalyzer :VBANdwidth :AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set whether the user or the instrument sets the video bandwidth for spectrum analyzer channels. If the instrument is automatically setting the video bandwidth then changing start/stop/centre/span frequencies may alter the video bandwidth.

Example: `:SAN:VBAN:AUTO ON`

The instrument sets the video bandwidth.

:SANalyzer :VBANdwidth :AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the user or the instrument sets the video bandwidth for spectrum analyzer channels.

Example: `:SAN:VBAN:AUTO?`

:SANalyzer :VBANdwidth :VALue

Parameters: <NUMERIC VALUE>

video bandwidth

Valid values: video bandwidth: real

Suffix: video bandwidth: The following suffixes are accepted for frequency: Hz, kHz and MHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the video bandwidth for spectrum analyzer channels. This will automatically switch into 'user set video bandwidth' mode (i.e. an implied :SAN:VBAN:AUTO OFF is sent first). Since video bandwidth is settable in discrete steps, the entered value will be rounded to the nearest valid value. See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SAN:VBAN:VAL 10E6`

Set a resolution bandwidth of 10 MHz.

:SANalyzer
:VBANdwidth
:VALue?

Parameters: none

Response: <NR2>

video bandwidth

Returned values: video bandwidth: real

Description: Determine the video bandwidth (in Hz) for the active spectrum analyzer channel. Note that this command will return the current value of video bandwidth regardless of whether it was set automatically or by the user.

Example:
```
:SAN:VBAN:VAL?
```

SCALar subsystem

```
SCALar
    GDELay
        APERture
            COUPling\?
            [VALue]\?
        FM
            DEViation\?
        RANGe\?
        ZERO\?
    PCAL
        ID\?
        OPEN
            MERGe
                [RLOSs]
                SEILoss
            [ONLY]
                [RLOSs]
                SEILoss
        SELect
        SHORt
            MERGe
                [RLOSs]
                SEILoss
            [ONLY]
                [RLOSs]
                SEILoss
        [STATe]\?
        THRough
            [NORMal]
            OFFSet
        TYPE?
        VERify?
    RX
        LEVel\?
        RBANdwidth\?
    SMOothing
        APERture\?
        [STATe]\?
```

:SCALar
:GDELay
:APERture
:COUPling

Parameters: <BOOLEAN PROGRAM DATA>

aperture coupling

Description: Couple or uncouple the aperture value to the span. When coupled, the instrument will recalculate the aperture value on any span change, and then calculate the FM deviation and range values from the aperture. When uncoupled, the user can enter an aperture value and the FM deviation and range will be determined from that value.

Example: `:SCAL:GDEL:APER:COUP OFF`

Switch aperture coupling off.

:SCALar
:GDELay
:APERture
:COUPling?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

aperture coupling

Description: Determine whether aperture is coupled to span or not.

Example: `:SCAL:GDEL:APER:COUP?`

:SCALar
:GDELay
:APERture
[:VALue]

Parameters: <NUMERIC VALUE>

group delay aperture

Valid values: Group delay aperture: real

Suffix: Group delay aperture: The following suffixes are accepted for frequency: Hz, kHz and MHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the aperture for group delay measurements in scalar channels. The aperture can only be set to a fixed set of values, any number entered will be rounded to the nearest acceptable value. Sending this command will also uncouple the aperture from the span. On any change to the aperture, the FM deviation and range are re-calculated.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SCAL:GDEL:APER 200KHz`

Set an aperture of 200 kHz.

:SCALar
:GDELay
:APERture
[:VALue]?

Parameters: none

Response: <NR2>

group delay aperture

Returned values: group delay aperture: real

Description: Determine the aperture (in Hz) for the active scalar channel.

Example: `:SCAL:GDEL:APER?`

:SCALar
:GDELay
:FM
:DEViation

Parameters: <NUMERIC VALUE>

FM deviation value

Valid values: FM deviation value real

Suffix: FM deviation value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Sets the FM deviation value to be used for group delay sweeps. Sending this command will also uncouple the aperture from the span. Note that this is a different parameter than SOUR:FM:DEV – that is only used for CW FM (non group delay).

Example: `:SCAL:GDEL:FM:DEV 1E6`

Set the FM deviation (for group delay sweeps) to 1 MHz.

:SCALar
:GDELay
:FM
:DEViation?

Parameters: none

Response: <NR2>

FM deviation value

Returned values: FM deviation value: real

Description: Determine the FM deviation value for the active channel (group delay sweeps only).

Example: `:SCAL:GDEL:FM:DEV?`

:SCALar :GDELay :RANGe

Parameters: <NUMERIC VALUE>

group delay range

Valid values: group delay range: real

Suffix: group delay range: The following suffixes are accepted for time: ps, ns, us, ms and s. If no suffix is entered then the default suffix of s is assumed.

Description: Set the range for group delay measurements in scalar channels. Sending this command will also uncouple the aperture from the span.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SCAL:GDEL:RANG 1us`

Set a range of 1 μs.

:SCALar :GDELay :RANGe?

Parameters: none

Response: <NR2>

group delay range

Returned values: group delay range: real

Description: Determine the range (in seconds) for the active scalar channel.

Example: `:SCAL:GDEL:RANG?`

:SCALar
:GDELay
:ZERO

Parameters: <CPD>

group delay autozeroing

Valid values: group delay autozeroing: [AUTO | ON | OFF]. Values other than those stated are rejected and an error generated.

Description: For group delay measurements, selects how autozeroing is performed above 3 GHz. Auto will let the instrument select the appropriate setting, on will always perform microwave point by point zeroing, and off will never perform microwave point by point zeroing.

Example: `:SCAL:GDEL:ZERO ON`

Force microwave point by point zeroing on above 3 GHz.

:SCALar
:GDELay
:ZERO?

Parameters: none

Response: <CRD>

group delay autozeroing

Returned values: group delay autozeroing: [AUTO | ON | OFF]

Description: Determine the mode of microwave zeroing for group delay measurements.

Example: `:SCAL:GDEL:ZERO?`

:SCALar :PCAL :ID

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name, user id

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

user id: string

Description: Set the user entered id string of the specified path calibration.

Example: `:SCAL:PCAL:ID "PCL1", "My special calibration"`

:SCALar :PCAL :ID?

Parameters: none

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the user entered id string of the path calibration associated with the active trace. The id string gives additional details of the path calibration store over what can be determined from the file name alone.

Example: `:SCAL:PCAL:ID?`

:SCALar
:PCAL
:OPEN
:MERGe
[:RLOSs]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'open' path calibration and apply the path calibration upon completion. If the specified store already exists and contains 'short' return loss data, the 'open' data will be merged with it, otherwise, an error is generated. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:OPEN:MERG "PCL2"`

Perform an 'open' path calibration and merge in path cal store PCL2.

:SCALar
:PCAL
:OPEN
:MERGe
:SEILoss

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'open' path calibration for a single-ended insertion loss measurement and apply the path calibration upon completion. If the specified store already exists and contains 'short' single ended insertion loss data, the 'open' data will be merged with it, otherwise, an error will be generated. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:OPEN:MERG:SEIL "PCL1"`

Perform an 'open' single ended insertion loss path calibration and merge in path cal store PCL1.

:SCALar
:PCAL
:SELect

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Assign a path calibration store to the active measurement. Path calibrations cannot be stored on the floppy disk. They may be copied from a floppy using MMEM:COPY. If the calibration store does not hold a valid scalar path calibration an error will be generated.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Example: `:SCAL:PCAL:SEL "PCL1"`

Assign path cal store PCL1 *to the measurement.*

:SCALar
:PCAL
:SHORt
:MERGe
[:RLOSs]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate a 'short' path calibration and apply the path calibration upon completion. If the specified store already exists and contains 'open' return loss data, the 'short' data will be merged with it, otherwise, an error is generated. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk.

Example: `:SCAL:PCAL:SHOR:MERG "PCL3"`

Perform a 'short' path calibration and merge in path cal store PCL3.

:SCALar
:PCAL
:SHORt
:MERGe
:SEILoss

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'short' path calibration for a single-ended insertion loss measurement and apply the path calibration upon completion. If the specified store already exists and contains 'open' single ended insertion loss data, the 'short' data will be merged with it, otherwise, an error will be generated. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:SHOR:MERG:SEIL "PCL2"`

Perform a 'short' single ended insertion loss path calibration and merge in path cal store PCL2.

:SCALar
:PCAL
:SHORt
[:ONLY]
[:RLOSs]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'short' path calibration, storing the data in the specified path calibration store, and apply the path calibration upon completion. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the operation status register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3, and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:SHOR "PCL1"`

Perform a short path calibration and store in path cal store PCL1.

:SCALar
:PCAL
:SHORt
[:ONLY]
:SEILoss

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'short' path calibration for a single-ended insertion loss measurement, storing the data in the specified path calibration store, and apply the path calibration upon completion. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:SHOR:SEIL "PCL4"`

Perform a short single ended insertion loss path calibration and store in path cal store PCL4.

:SCALar
:PCAL
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable path calibration

Example: `:SCAL:PCAL OFF`

Switch path calibration off. This is the normal use for the command, since saving a measurement to a path cal memory enables path calibration automatically.

:SCALar :PCAL [:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether a path calibration is enabled.

Example: `:SCAL:PCAL?`

:SCALar :PCAL :THRough [NORMal]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'through' path calibration, storing the data in the specified path calibration store, and apply the path calibration upon completion. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command should be used at all times except when the offset source is being used for a scalar RX measurement. In that case use :SCAL:PCAL:THR:OFFS instead.

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the Operation Status Register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3 and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:THR "PCL2"`

Perform an through path calibration and store in path cal store PCL2.

:SCALar
:PCAL
:THRough
:OFFSet

Parameters: <BOOLEAN PROGRAM DATA>, <STRING PROGRAM DATA>

cal over source range, file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Initiate an 'through' path calibration, storing the data in the specified path calibration store, and apply the path calibration upon completion. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This command should only be used when using an offset source in a scalar channel. The first parameter is used to perform the calibration over the source frequency range (1) or over the display frequency range (0).

This command is an overlapped command and (as long as sweeps are not being triggered using *TRG), *OPC, *OPC?, or *WAI can be used to determine when the calibration has completed. If the calibration is being performed in triggered mode then it is necessary to read bit 0 of the operation status register to determine when the calibration has completed.

Any command received whilst the calibration is in progress that cannot be actioned because it would affect the calibration will generate an error.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3, and pcl4. Entering any filename other than one of these four will give an error.

Path calibrations cannot be directly stored on the floppy disk. They may be copied to a floppy using MMEM:COPY.

Example: `:SCAL:PCAL:THR:OFFS 1, "PCL2"`

Perform an through path calibration using source frequency range and store in path cal store PCL2.

:SCALar :PCAL :TYPE?

Parameters: none

Response: <CRD>, <BOOLEAN RESPONSE DATA>

path calibration type, single ended insertion loss

Returned values: path calibration type: [OPEN | SHOR | BOTH | THR]

Description: Determine the type of the path calibration assigned to the active trace. The returned values are:

cal type	single ended	
OPEN	false (0)	Open return loss path calibration
SHOR	false (0)	Short return loss path calibration
BOTH	false (0)	Short/Open return loss path calibration
THR	false (0)	Through path calibration
OPEN	true (1)	Open single ended insertion loss path calibration
SHOR	true (1)	Short single ended insertion loss path calibration
BOTH	true (1)	Short/Open single ended insertion loss path calibration
NONE	false (0)	No path calibration applied

Example:
```
:SCAL:PCAL:TYPE?
```

:SCALar :PCAL :VERify?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <BOOLEAN RESPONSE DATA>

valid scalar path cal

Description: Determine if the specified calibration store contains a scalar path calibration. Since calibration stores are capable of holding either scalar path cals or fault location cals, this allows the user to confirm that a store actually contains scalar path calibration data.

There are four fixed filenames that can be used to store either fault location calibrations or scalar path calibrations. These names are pcl1, pcl2, pcl3, and pcl4. Entering any filename other than one of these four will give an error.

Example:
```
:SCAL:PCAL:VER? "PCL3"
```

:SCALar
:RX
:LEVel

Parameters: <NUMERIC VALUE>

operating level

Valid values: operating level: real

Suffix: operating level: A suffix of dBm is accepted for power. If no suffix is entered then the default suffix of dBm is assumed.

Description: Set the operating level for RX measurements or group delay measurements on scalar channels.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SCAL:RX:LEV -10`

Set the operating level for RX measurements to -10 dBm.

:SCALar
:RX
:LEVel?

Parameters: none

Response: <NR2>

operating level

Returned values: operating level: real

Description: Determine the operating level for the active channel.

Example: `:SCAL:RX:LEV?`

:SCALar
:RX
:RBANdwidth

Parameters: <NUMERIC VALUE>

resolution bandwidth

Valid values: resolution bandwidth: real

Suffix: resolution bandwidth: The following suffixes are accepted for frequency: Hz, kHz and MHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the resolution bandwidth for RX measurements in scalar channels. Since resolution bandwidth is settable in discrete steps, the entered value will be rounded to the nearest valid value. See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SCAL:RX:RBAN 3E6`

Set a resolution bandwidth of 3 MHz.

:SCALar
:RX
:RBANdwidth?

Parameters: none

Response: <NR2>

resolution bandwidth

Returned values: resolution bandwidth: real

Description: Determine the resolution bandwidth (in Hz) for the active scalar channel.

Example: `:SCAL:RX:RBAN?`

:SCALar :SMOothing :APERture

Parameters: <NUMERIC VALUE>

aperture

Valid values: aperture: real. Valid values are 1 to 20. Values outside range are clipped.

Suffix: aperture: A suffix of pct is accepted for a value of percent. If no suffix is entered then the default suffix of pct is assumed.

Description: Set the smoothing aperture. The value is a percentage of the domain span.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SCAL:SMO:APER 4`

Set the smoothing aperture to 4%.

:SCALar :SMOothing :APERture?

Parameters: none

Response: <NR2>

aperture

Returned values: aperture: real. Valid values are 1 to 20.

Description: Determine the smoothing aperture for the active measurement.

Example: `:SCAL:SMO:APER?`

:SCALar :SMOothing [:STATe]

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable smoothing for the active measurement.

Example: `:SCAL:SMO ON`

Switch smoothing on.

:SCALar
:SMOothing
[:STATe]?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether smoothing is enabled for the active measurement.

Example:
```
:SCAL:SMO?
```

SOURce subsystem

SOURce
- BLANking\?
- CALibration
  - FM
  - FREQuency
  - POWer
    - [BBANd]
    - NBANd
  - SELect\?
  - STANdard
  - TRANsfer
- DOMain
  - [ASCii]?
  - BINary?
- FM
  - [DEViation]\?
  - EXTernal
    - COUPling\?
    - DCNull
  - INTernal
    - FREQuency\?
  - SOURce\?
  - STATe\?
- FREQuency
  - CENTer\?
  - CENTre\?
  - [CW]\?
  - SPAN\?
  - STANdard\?
  - STARt\?
  - STOP\?
- LEVelling\?
- MODE\?
- POWer
  - LEVel\?
  - STARt\?
  - STOP\?
- RF\?
- SBANdwidth
  - MICRowave\?
  - MODE\?
  - RF\?
- SWEep
  - AUTO\?
  - POINts\?
  - TIME\?
- VOUTput
  - MODE\?
  - VALue\?

:SOURce
:BLANking

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set source RF blanking on or off.

Example: `:SOUR:BLAN ON`

Set source RF blanking on.

:SOURce
:BLANking?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether source RF blanking is on or off.

Example: `:SOUR:BLAN?`

:SOURce
:CALibration
:FM

Parameters: none

Description: Perform a source FM calibration and save data to source FM calibration store. The data is used immediately by the instrument. Any previous FM calibration data is lost.

Example: `:SOUR:CAL:FM`

:SOURce
:CALibration
:FREQuency

Parameters: none

Description: Perform a source frequency calibration and save data to source frequency calibration store. The data is used immediately by the instrument. Any previous frequency calibration data is lost.

Example: `:SOUR:CAL:FREQ`

:SOURce
:CALibration
:POWer
[:BBANd]

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Perform a broad band source power calibration and save data to specified user source power calibration store. The store will be accessed using the file name sent as part of this command. A file extension should not be specified as this is fixed by the instrument. Source power calibrations cannot be saved to the floppy disk.

The user store can then be selected or the data can be transferred to the primary power calibration store.

If the user source power calibration store being saved to is the one currently in use then the data will be used immediately, otherwise it will only be used when the store is selected, or, if the primary power cal is in use, when the user store is transferred to the primary power cal.

Example: `:SOUR:CAL:POW "MY_CAL"`

Perform a broad band source power cal and save to user power calibration store MY_CAL.

:SOURce
:CALibration
:POWer
:NBANd

Parameters: <NUMERIC VALUE>, <NUMERIC VALUE>, <STRING PROGRAM DATA>

start frequency, stop frequency, file name

Valid values: start frequency: real
stop frequency: real
file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Suffix: start frequency: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

stop frequency: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Perform a narrow band source power calibration and save data to specified user source power calibration store. The store will be accessed using the file name sent as part of this command. A file extension should not be specified as this is fixed by the instrument. Source power calibrations cannot be saved to the floppy disk.

The user store can then be selected or the data can be transferred to the primary power calibration store.

If the user source power calibration store being saved to is the one currently in use then the data will be used immediately, otherwise it will only be used when the store is selected, or, if the primary power cal is in use, when the user store is transferred to the primary power cal.

Example: `:SOUR:CAL:POW:NBAN 2GHZ, 5.5GHZ, "MY_CAL"`

Perform a narrow band source power cal and save to user power calibration store MY_CAL.

:SOURce
:CALibration
:SELect

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Select a source power calibration. If the filename specified is PRIMPWR then the primary power calibration will be selected otherwise the specified user power calibration will be selected.

Example: `:SOUR:CAL:SEL "MY_CAL"`

Select user source power calibration MY_CAL.

:SOURce :CALibration :SELect?

Parameters: none

Response: <STRING RESPONSE DATA>

calibration file name

Returned values: calibration file name: string

Description: Determine which power calibration is in use. If the primary power calibration is selected, this command will return PRIMPWR, otherwise the name of the selected user power calibration is returned.

Example:
```
:SOUR:CAL:SEL?
```

:SOURce :CALibration :STANdard

Parameters: none

Description: Perform a source frequency standard calibration and save data to source frequency standard calibration store. The data is used immediately by the instrument. Any previous frequency standard calibration data is lost.

Example:
```
:SOUR:CAL:STAN
```

:SOURce :CALibration :TRANsfer

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Transfer a user source power calibration to the primary power calibration.

Example:
```
:SOUR:CAL:TRAN "FULL"
```

Transfer user source power cal FULL *to the primary source power calibration.*

:SOURce
:DOMain
[:ASCii]?

Parameters: none

Response: <NR3>, ..., <NR3>

domain at point 0, ..., domain at point *n*-1

Returned values: domain at point 0: real
...
domain at point *n*-1: real

Description: Read the domain values for every source point. Note that these are measured points not display points, for instance, in a fault location channel the values returned will be the frequencies that the source is stepping. This command is also valid in source only mode.

Example: `:SOUR:DOM?`

:SOURce
:DOMain
:BINary?

Parameters: none

Response: <DEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

domain data

Description: Read the domain values for every source point. Note that these are measured points not display points, for instance, in a fault location channel the values returned will be the frequencies that the source is stepping. This command is also valid in source only mode.

The returned data is organised as follows:

Each domain value consists of 8 bytes received in the order:
byte 0 : S E E E E E E E
byte 1 : E E E E F F F F
byte 2 : F F F F F F F F
byte 3 : F F F F F F F F
byte 4 : F F F F F F F F
byte 5 : F F F F F F F F
byte 6 : F F F F F F F F
byte 7 : F F F F F F F F
Where S is sign bit, E is exponent (11 bits), and F is fractional part (52 bits).

These bytes hold a 64 bit (IEEE single precision) number conforming to the IEEE Standard of Binary Floating-Point Arithmetic, ANSI/IEEE Std 754-1985.

Example: `:SOUR:DOM:BIN?`

:SOURce
:FM
[:DEViation]

Parameters: <NUMERIC VALUE>

FM deviation value

Valid values: FM deviation value real

Suffix: FM deviation value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar or spectrum analyzer channel, this sets the source FM deviation value. This command is invalid if the domain is not frequency.

For a fault location channel, this command will give an error.

This command is also valid in source only mode.

This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM 1E6`

Set the FM deviation to 1 MHz.

:SOURce
:FM
[:DEViation]?

Parameters: none

Response: <NR2>

FM deviation value

Returned values: FM deviation value: real

Description: Determine the source FM deviation value for the active channel. This command is also valid in source only mode.

This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM?`

:SOURce :FM :EXTernal :COUPling

Parameters: <CPD>

external coupling

Valid values: external coupling: [AC | DC]. Values other than those stated are rejected and an error generated.

Description: Select whether the external FM signal is AC or DC coupled. This command is also valid in source only mode.

This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM:EXT:COUP AC`

Select AC coupling of the FM signal.

:SOURce :FM :EXTernal :COUPling?

Parameters: none

Response: <CRD>

external coupling

Returned values: external coupling: [AC | DC]

Description: Determine whether DC or AC coupling is selected. This command is also valid in source only mode.

This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM:EXT:COUP?`

:SOURce :FM :EXTernal :DCNull

Parameters: none

Description: Perform a DC null operation. The FM source should be external DC.

This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM:EXT:DCN`

Perform a DC Null. External DC FM must be selected.

:SOURce
:FM
:INTernal
:FREQuency

Parameters: <NUMERIC VALUE>

internal frequency value

Valid values: internal frequency value: real

Suffix: internal frequency value: The following suffixes are accepted for frequency: Hz, kHz and MHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: This sets the internal modulation frequency to be used for CW FM. For the value to be used the internal mod generator must be selected using SOUR:FM:SOUR INT.

This command is only valid if the group delay option is present.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:FM:INT:FREQ 200E3`

Set the internal mod frequency to 200 kHz.

:SOURce
:FM
:INTernal
:FREQuency?

Parameters: none

Response: <NR2>

internal frequency value

Returned values: internal frequency value: real

Description: Determine the internal mod frequency value for the active channel. This command is only valid if the group delay option is fitted. This command is also valid in source only mode.

Example: `:SOUR:FM:INT:FREQ?`

:SOURce
:FM
:SOURce

Parameters: <CPD>

coupling mode

Valid values: coupling mode: [INTernal | EXTernal]. Values other than those stated are rejected and an error generated.

Description: Select whether the internal modulator or an external frequency source is being used to generate FM. When selecting external, external AC will be selected. To change to external DC use the SOUR:FM:EXT:COUP DC command. This command is also valid in source only mode.

This command is only valid if the group delay option is fitted to the instrument.

Example: `:SOUR:FM:SOUR EXT`

Select external (AC) modulation source for generation of the FM signal.

:SOURce
:FM
:SOURce?

Parameters: none

Response: <CRD>

coupling mode

Returned values: coupling mode: [INT | EXT]

Description: Determine whether internal or external modulation is selected. This command is also valid in source only mode. This command is only valid if the group delay option is fitted to the instrument.

Example: `:SOUR:FM:SOUR?`

:SOURce
:FM
:STATe

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set FM on or off. This command is also valid in source only mode.

This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM:STAT ON`

Set FM on.

:SOURce :FM :STATe?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether FM is on. This command is also valid in source only mode. This command is only valid if the FM option is fitted to the instrument.

Example: `:SOUR:FM:STAT?`

:SOURce :FREQuency :CENTer

Parameters: <NUMERIC VALUE>

frequency centre value

Valid values: frequency centre value: real

Suffix: frequency centre value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel, this sets the source frequency centre value. This command is invalid if the domain is not frequency.

For a fault location channel, this command is only valid for range entry mode, when it will set the source centre frequency.

For a spectrum analyzer channel, this command will give an error.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:FREQ:CENT 15E9`

Set the centre frequency to 15 GHz.

:SOURce :FREQuency :CENTer?

Parameters: none

Response: <NR2>

frequency centre value

Returned values: frequency centre value: real

Description: Determine the source frequency centre value for the active channel. Note that this command is invalid if the instrument is set up such that the user cannot set the centre value. This command is also valid in source only mode.

Example:
```
:SOUR:FREQ:CENT?
```

:SOURce :FREQuency :CENTre

Parameters: <NUMERIC VALUE>

frequency centre value

Valid values: frequency centre value: real

Suffix: frequency centre value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel, this sets the source frequency centre value. Since for scalar channels the display start/stop is identical to source start/stop, this command is identical to :DISP:CENT. This command is invalid if the domain is not frequency.

For a fault location channel, this command is only valid for range entry mode, when it will set the source centre frequency.

For a spectrum analyzer channel, this command will give an error.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:SOUR:FREQ:CENT 15E9
```

Set the centre frequency to 15 GHz.

:SOURce :FREQuency :CENTre?

Parameters: none

Response: <NR2>

frequency centre value

Returned values: frequency centre value: real

Description: Determine the source frequency centre value for the active channel. Note that this command is invalid if the instrument is set up such that the user cannot set the centre value. This command is also valid in source only mode.

Example: `:SOUR:FREQ:CENT?`

:SOURce :FREQuency :CW

Parameters: <NUMERIC VALUE>

CW frequency

Valid values: CW frequency: real

Suffix: CW frequency: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel, this sets the CW frequency. This value is only used when the source is set to CW mode or a power sweep using :SOUR:MODE.

This command is invalid for a fault location channel.

For a spectrum analyzer channel, this command will set the source frequency if the source is in a CW mode. If the source is set in tracking generator mode then this command will give an error.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:FREQ:CW 15E9`

Set the CW frequency to 15 GHz.

:SOURce :FREQuency :CW?

Parameters: none

Response: <NR2>

CW frequency

Returned values: CW frequency: real

Description: Determine the CW frequency. This command is also valid in source only mode.

Example: `:SOUR:FREQ:CW?`

:SOURce :FREQuency :SPAN

Parameters: <NUMERIC VALUE>

frequency span value

Valid values: frequency span value: real

Suffix: frequency span value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel, this sets the frequency span (the :SOUR:MODE command is used to specify the domain as frequency).

This command is invalid for a fault location channel.

For a spectrum analyzer channel, this command will give an error.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:FREQ:SPAN 10E9`

Set the frequency span to 10 GHz (assuming the domain is frequency).

:SOURce :FREQuency :SPAN?

Parameters: none

Response: <NR2>

span value

Returned values: span value: real

Description: Determine the source frequency span. This command is also valid in source only mode.

Example:
```
:SOUR:FREQ:SPAN?
```

:SOURce :FREQuency :STANdard

Parameters: <CPD>

frequency standard

Valid values: frequency standard: [INT | EX1 | EX10]. Values other than those stated are rejected and an error generated.

Description: Select a source frequency standard: internal, external 1 MHz or external 10 MHz. This command is also valid in source only mode.

Example:
```
:SOUR:FREQ:STAN EX1
```

Select the rear panel FREQ STD INPUT/OUTPUT to accept a 1 MHz external frequency standard.

:SOURce :FREQuency :STANdard?

Parameters: none

Response: <CRD>

frequency standard

Returned values: frequency standard: [INT | EX1 | EX10]

Description: Determine the source frequency standard in use. This command is also valid in source only mode.

Example:
```
:SOUR:FREQ:STAN?
```

:SOURce
:FREQuency
:STARt

Parameters: <NUMERIC VALUE>

frequency start value

Valid values: frequency start value: real

Suffix: frequency start value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel, this sets the source frequency start value (the :SOUR:MODE command is used to specify the domain).

For a fault location channel, if the entry mode is frequency, this command sets the source start frequency. Use :FLOC:ENTR FREQ to select frequency entry mode.

For a spectrum analyzer channel, this command will give an error.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:FREQ:STAR 10.0E9`

Set the start frequency to 10 GHz.

:SOURce
:FREQuency
:STARt?

Parameters: none

Response: <NR2>

frequency start value

Returned values: frequency start value: real

Description: Determine the source frequency start value. This command is also valid in source only mode.

Example: `:SOUR:FREQ:STAR?`

:SOURce :FREQuency :STOP

Parameters: <NUMERIC VALUE>

frequency stop value

Valid values: frequency stop value: real

Suffix: frequency stop value: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: For a scalar channel, this sets the source frequency stop value (the :SOUR:MODE command is used to specify the domain).

For a fault location channel, if the entry mode is frequency, this command sets the source stop frequency. Use :FLOC:ENTR FREQ to select frequency entry mode.

For a spectrum analyzer channel, this command will give an error.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:FREQ:STOP 16.5E9`

Set the stop frequency to 16.5 GHz (assuming the domain is frequency).

:SOURce :FREQuency :STOP?

Parameters: none

Response: <NR2>

frequency stop value

Returned values: frequency stop value: real

Description: Determine the source frequency stop value. This command is also valid in source only mode.

Example: `:SOUR:FREQ:STOP?`

:SOURce
:LEVelling

Parameters: <CPD>

levelling mode

Valid values: levelling mode: [INTernal | POSitive | NEGative | PMETer]. Values other than those stated are rejected and an error generated.

Description: Set the source levelling mode:

INTernal	Internal levelling
POSitive	External levelling - using a +ve polarity detector
NEGative	External levelling - using a -ve polarity detector
PMETer	External levelling - using a power meter

This command is also valid in source only mode.

Example: `:SOUR:LEV PMET`

Enable power meter levelling.

:SOURce
:LEVelling?

Parameters: none

Response: <CRD>

levelling mode

Returned values: levelling mode: [INT | POS | NEG | PMET]

Description: Determine the source levelling mode. This command is also valid in source only mode.

Example: `:SOUR:LEV?`

:SOURce :MODE

Parameters: <CPD>

source mode

Valid values: source mode: [POWer | FREQuency | CW]. Values other than those stated are rejected and an error generated.

Description: Set the source mode for the active channel. If channel coupling is enabled then the inactive channel will also be changed.

This command generates an error on fault location and spectrum analyzer channels since the source mode cannot be directly set by the user.

Valid source modes are:

POWer	Linear power sweep
FREQuency	Linear frequency sweep
CW	CW output

This command is also valid in source only mode.

Example: `:SOUR:MODE POW`

Set power sweep mode.

:SOURce :MODE?

Parameters: none

Response: <CRD>

source mode

Returned values: source mode: [POW | FREQ | CW]

Description: Determine the source mode for the active channel. This command is also valid in source only mode.

Example: `:SOUR:MODE?`

:SOURce
:POWer
:LEVel

Parameters: <NUMERIC VALUE>

power level

Valid values: power level: real

Suffix: power level: A suffix of dBm is accepted for power. If no suffix is entered then the default suffix of dBm is assumed.

Description: For a scalar channel this sets the source output power. This value is only used when the source is set to a frequency sweep mode or a CW frequency mode using :SOUR:MODE.

For a fault location channel this sets the source output power.

For a spectrum analyzer channel, this command will set the source output power.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:POW:LEV 1.0`

Set output power to 1 dBm.

:SOURce
:POWer
:LEVel?

Parameters: none

Response: <NR2>

power level

Returned values: power level: real

Description: Determine the source output power. This command is also valid in source only mode.

Example: `:SOUR:POW:LEV?`

:SOURce
:POWer
:STARt

Parameters: <NUMERIC VALUE>

power start value

Valid values: power start value: real

Suffix: power start value: A suffix of dBm is accepted for power. If no suffix is entered then the default suffix of dBm is assumed.

Description: For a scalar channel, this sets the source power start value (the :SOUR:MODE command is used to specify the domain). This command is invalid if the domain is not power.

For a fault location channel this command is invalid.

For a spectrum analyzer channel this command is invalid.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:POW:STAR 5`

Set the start power to 5dB.

:SOURce
:POWer
:STARt?

Parameters: none

Response: <NR2>

power start value

Returned values: power start value: real

Description: Determine the source power start value. This command is invalid if the domain is not power. This command is also valid in source only mode.

Example: `:SOUR:POW:STAR?`

:SOURce
:POWer
:STOP

Parameters: <NUMERIC VALUE>

power stop value

Valid values: power stop value: real

Suffix: power stop value: A suffix of dBm is accepted for power. If no suffix is entered then the default suffix of dBm is assumed.

Description: For a scalar channel, this sets the source power stop value (the :SOUR:MODE command is used to specify the domain). This command is invalid if the domain is not power.

For a fault location channel this command is invalid.

For a spectrum analyzer channel this command is invalid.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:POW:STOP 10`

Set the stop power to 10 dB.

:SOURce
:POWer
:STOP?

Parameters: none

Response: <NR2>

power stop value

Returned values: power stop value: real

Description: Determine the source power stop value. This command is also valid in source only mode.

Example: `:SOUR:POW:STOP?`

:SOURce :RF

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set RF output on or off. This command is also valid in source only mode.

Example: `:SOUR:RF ON`

Set RF on.

:SOURce :RF?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether RF output is on. This command is also valid in source only mode.

Example: `:SOUR:RF?`

:SOURce :SBANdwidth :MICRowave

Parameters: <CPD>

loop bandwidth state

Valid values: loop bandwidth state: [FAST | SLOW]. Values other than those stated are rejected and an error generated.

Description: Set the loop bandwidth state for the source output block. This command is also valid in source only mode.

This value will only be used if the loop bandwidth control has been set to manual using SOUR:SBAN:MODE MAN.

Example: `:SOUR:SBAN:MICR FAST`

Set the loop bandwidth state to fast settling mode.

:SOURce :SBANdwidth :MICRowave?

Parameters: none

Response: <CRD>

loop bandwidth state

Returned values: loop bandwidth state: [FAST | SLOW]

Description: Determine the synthesiser loop bandwidth state for the source output block. This command is also valid in source only mode.

This command will always return the setting that is being used by the hardware.

Example: `:SOUR:SBAN:MICR?`

:SOURce :SBANdwidth :MODE

Parameters: <CPD>

loop bandwidth mode

Valid values: loop bandwidth mode: [AUTO | MANual]. Values other than those stated are rejected and an error generated.

Description: Set the loop bandwidth mode. This command is also valid in source only mode.

Setting AUTO mode will make the instrument determine the appropriate values for the RF and microwave loop bandwidth settings. Values set using SOUR:SBAN:RF and SOUR:SBAN:MICR will be ignored.

Setting MANual mode enables the two commands (SOUR:SBAN:RF and SOUR:SBAN:MICR) so that the user can specify how the loop bandwidth should be set.

Example: `:SOUR:SBAN:MODE AUTO`

Set the loop bandwidth mode to automatic.

:SOURce :SBANdwidth :MODE?

Parameters: none

Response: <CRD>

loop bandwidth mode

Returned values: loop bandwidth mode: [AUTO | MAN]

Description: Determine the loop bandwidth mode. This command is also valid in source only mode.

Example: `:SOUR:SBAN:MODE?`

:SOURce
:SBANdwidth
:RF

Parameters: <CPD>

loop bandwidth state

Valid values: loop bandwidth state: [FAST | SLOW]. Values other than those stated are rejected and an error generated.

Description: Set the loop bandwidth state for the core block. This command is also valid in source only mode.

This value will only be used if the loop bandwidth control has been set to manual using SOUR:SBAN:MODE MAN.

Example: `:SOUR:SBAN:RF FAST`

Set the loop bandwidth state to fast settling mode.

:SOURce
:SBANdwidth
:RF?

Parameters: none

Response: <CRD>

loop bandwidth state

Returned values: loop bandwidth state: [FAST | SLOW]

Description: Determine the loop bandwidth state for the core block. This command is also valid in source only mode.

This command will always return the setting that is being used by the hardware.

Example: `:SOUR:SBAN:RF?`

:SOURce
:SWEep
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable/disable automatic setting of sweep time for the active channel. This command does not apply to a spectrum analyzer channel.
If channel coupling is on, the non active channel is also affected.

This command is also valid in source only mode.

Example: `:SOUR:SWE:AUTO OFF`

Enable sweep time to be set by the user.

:SOURce :SWEep :AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether sweep time is set automatically for the active channel. This command is also valid in source only mode.

Example:
```
:SOUR:SWE:AUTO?
```

:SOURce :SWEep :POINts

Parameters: <NUMERIC VALUE>

number of points

Valid values: number of points: integer

Suffix: number of points: No suffix is allowed.

Description: Set the number of measurement points for the active channel. The number of points will be clipped as necessary.

This command is valid for scalar and fault location channels only.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:SOUR:SWE:POIN 201
```

Set number of points to 201.

:SOURce :SWEep :POINts?

Parameters: none

Response: <NR1>

number of points

Returned values: number of points: integer

Description: Determine the number of source sweep points for the active channel.

This command is valid for scalar and fault location channels only.

This command is also valid in source only mode.

Example:
```
:SOUR:SWE:POIN?
```

:SOURce :SWEep :TIME

Parameters: <NUMERIC VALUE>

sweep time

Valid values: sweep time: real

Suffix: sweep time: A suffix of either ms or s is accepted for time. If no suffix is entered then the default suffix of s is assumed.

Description: Set the sweep time for the active channel. This command applies to all channel types except spectrum analyzer, use SAN:SWE:VAL instead. If channel coupling is on, the non active channel is also affected. Setting a user sweep time will disable the automatic sweep time function.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:SOUR:SWE:TIME 0.5
```

Set a sweep time of 500 ms.

:SOURce :SWEep :TIME?

Parameters: none

Response: <NR2>

sweep time

Returned values: sweep time: real

Description: Determine the sweep time (in seconds) for the active channel. Note that this command will return the current value of sweep time regardless of whether it was set automatically or by the user. This command is also valid in source only mode.

Example:
```
:SOUR:SWE:TIME?
```

:SOURce :VOUTput :MODE

Parameters: <CPD>

voltage output mode

Valid values: voltage output mode: [VRAMp | YAXis | FVOLtage]. Values other than those stated are rejected and an error generated.

Description: Set the mode for the VOLTAGE OUTPUT located on the rear panel. The available modes are:

VRAMp	voltage ramp (0-20 V)
YAXis	live Y-axis. Voltage follows active marker response value
FVOLtage	Fixed voltage, user settable value on a per channel basis

This command is also valid in source only mode.

Example:
```
:SOUR:VOUT:MODE VRAM
```

Set the VOLTAGE OUTPUT to voltage ramp mode.

:SOURce :VOUTput :MODE?

Parameters: none

Response: <CRD>

voltage output mode

Returned values: voltage output mode: [VRAM | YAX | FVOL]

Description: Read the VOLTAGE OUTPUT mode. This command is also valid in source only mode.

Example:
```
:SOUR:VOUT:MODE?
```

:SOURce :VOUTput :VALue

Parameters: <NUMERIC VALUE>

voltage

Valid values: voltage: real

Suffix: voltage: A suffix of either mV or V is accepted for voltage. If no suffix is entered then the default suffix of V is assumed.

Description: Set the fixed voltage value for the voltage output when the mode is set to fixed voltage mode. The value is settable on a per channel basis.

This command is also valid in source only mode.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:SOUR:VOUT:VAL 2`

Set the voltage output fixed voltage to 2V.

:SOURce :VOUTput :VALue?

Parameters: none

Response: <NR2>

voltage

Returned values: voltage: real

Description: Determine the fixed voltage. This command is also valid in source only mode.

Example: `:SOUR:VOUT:VAL?`

STATus subsystem

STATus
- OPERation
  - CONDition?
  - ENABle\?
  - [EVENt]?
  - NTRansition\?
  - PTRansition\?
- PRESet
- QUEStionable
  - CONDition?
  - ENABle\?
  - [EVENt]?
  - NTRansition\?
  - PTRansition\?

:STATus
:OPERation
:CONDition?

Parameters: none

Response: <NR1>

register contents

Returned values: register contents: integer. Values are in the range 0 to 32767.

Description: Read the contents of the Operation Status Condition Register. This register returns the current state of the instrument. Reading the register does not affect its contents. This is a sixteen bit register.

The meaning of each bit in the Operation Status Condition Register is given in Appendix A.

Example:
```
:STAT:OPER:COND?
```

:STATus
:OPERation
:ENABle

Parameters: <NRf>

mask

Valid values: mask: integer. Valid values are 0 to 65535. Values outside range are rejected and an error generated.

Description: Sets the enable mask which allows true conditions in the Operation Status Event Register to be reported in the summary bit (bit 7 in the Status Byte Register).

If a bit is 1 in the Operation Status Enable Register and its associated event bit (in the Operation Status Event Register) makes a transition to true, a positive transition will occur in the associated summary bit if that bit was previously 0.

Bit 15 of the mask value supplied is ignored since bit 15 of the Operation Status Enable Register is always zero. This is a sixteen bit register.

See Appendix A for more details

Example:
```
:STAT:OPER:ENAB 32
```

Program the mask associated with the Operation Status Event Register with the value 32 (0000 0000 0010 0000 in binary) to enable a positive transition in the summary bit when the instrument is waiting for trigger.

:STATus :OPERation :ENABle?

Parameters: none

Response: <NR1>

mask

Returned values: mask: integer. Values are in the range 0 to 32767.

Description: Read the mask from the Operation Status Enable Register. This is a sixteen bit register.

Example:
```
:STAT:OPER:ENAB?
```

:STATus :OPERation [:EVENt]?

Parameters: none

Response: <NR1>

event register contents

Returned values: event register contents: integer. Values are in the range 0 to 32767.

Description: Read the contents of the Operation Status Event Register. Reading the register will clear it. This is a sixteen bit register.

See Appendix A for more details

Example:
```
:STAT:OPER?
```

:STATus
:OPERation
:NTRansition

Parameters: <NRf>

negative transition mask

Valid values: negative transition mask: integer. Valid values are 0 to 65535. Values outside range are rejected and an error generated.

Description: Sets the negative transition filter which allows transitions from 1 to 0 in the Operation Status Condition Register to be latched into the Operation Status Event Register.

Bit 15 of the mask value supplied is ignored since bit 15 of the Operation Status Negative Transition Filter Register is always zero. This is a sixteen bit register.

See Appendix A for more details

Example: `:STAT:OPER:NTR 256`

Program the negative transition filter associated with the Operation Status Register with the value 256 (0000 0001 0000 0000 in binary) to enable the event register to be set when the instrument has reached the averaging target on channel one trace one.

:STATus
:OPERation
:NTRansition?

Parameters: none

Response: <NR1>

negative transition mask

Returned values: negative transition mask: integer. Values are in the range 0 to 32767.

Description: Read the mask from the Operation Status Negative Transition Filter Register. This is a sixteen bit register.

Example: `:STAT:OPER:NTR?`

:STATus :OPERation :PTRansition

Parameters: <NRf>

positive transition mask

Valid values: positive transition mask: integer. Valid values are 0 to 65535. Values outside range are rejected and an error generated.

Description: Sets the positive transition filter which allows transitions from 0 to 1 in the Operation Status Condition Register to be latched into the Operation Status Event Register.

Bit 15 of the mask value supplied is ignored since bit 15 of the Operation Status Positive Transition Filter Register is always zero. This is a sixteen bit register.

See Appendix A for more details

Example: `:STAT:OPER:PTR 32`

Program the positive transition filter associated with the Operation Status Register with the value 32 (0000 0000 0010 0000 in binary) to enable the event register to be set when the instrument is waiting for a trigger.

:STATus :OPERation :PTRansition?

Parameters: none

Response: <NR1>

positive transition mask

Returned values: positive transition mask: integer. Values are in the range 0 to 32767.

Description: Read the mask from the Operation Status Positive Transition Filter Register. This is a sixteen bit register.

Example: `:STAT:OPER:PTR?`

:STATus :PRESet

Parameters: none

Description: Preset the Operation Status Enable Register and the Questionable Status Enable Register to zero.

Also defaults the Operation status transition filters and the Questionable status transition filters, see Appendix A for more details

Example: `:STAT:PRES`

:STATus
:QUEStionable
:CONDition?

Parameters: none

Response: <NR1>

register contents

Returned values: register contents: integer. Values are in the range 0 to 32767.

Description: Read the contents of the Questionable Status Condition Register. This register returns the current state of the instrument. Reading the register does not affect its contents. This is a sixteen bit register.

The meaning of each bit in the Questionable Status Condition Register is given in Appendix A.

Example:
```
:STAT:QUES:COND?
```

:STATus
:QUEStionable
:ENABle

Parameters: <NRf>

mask

Valid values: mask: integer. Valid values are 0 to 65535. Values outside range are rejected and an error generated.

Description: Sets the enable mask which allows true conditions in the Questionable Status Event Register to be reported in the summary bit (bit 3 in the Status Byte Register).

If a bit is 1 in the Questionable Status Enable Register and its associated event bit (in the Questionable Status Event Register) makes a transition to true, a positive transition will occur in the associated summary bit if that bit was previously 0.

Bit 15 of the mask value supplied is ignored since bit 15 of the Questionable Status Enable Register is always zero. This is a sixteen bit register.

See Appendix A for more details

Example:
```
:STAT:QUES:ENAB 8
```

Program the mask associated with the Questionable Status Event Register with the mask value 8 (0000 0000 0000 1000 in binary) to enable a positive transition in the summary bit when the source becomes unlevelled.

:STATus :QUEStionable :ENABle?

Parameters: none

Response: <NR1>

mask

Returned values: mask: integer. Values are in the range 0 to 32767.

Description: Read the mask from the Questionable Status Enable Register. This is a sixteen bit register.

Example:
```
:STAT:QUES:ENAB?
```

:STATus :QUEStionable [:EVENt]?

Parameters: none

Response: <NR1>

event register contents

Returned values: event register contents: integer. Values are in the range 0 to 32767.

Description: Read the contents of the Questionable Status Event Register. Reading the register will clear it. This is a sixteen bit register.

See Appendix A for more details

Example:
```
:STAT:QUES?
```

:STATus
:QUEStionable
:NTRansition

Parameters: <NRf>

negative transition mask

Valid values: negative transition mask: integer. Valid values are 0 to 65535. Values outside range are rejected and an error generated.

Description: Sets the negative transition filter which allows transitions from 1 to 0 in the Questionable Status Condition Register to be latched into the Questionable Status Event Register.

Bit 15 of the mask value supplied is ignored since bit 15 of the Questionable Status Negative Transition Filter Register is always zero. This is a sixteen bit register.

See Appendix A for more details

Example: `:STAT:QUES:NTR 256`

Program the negative transition filter associated with the Questionable Status Register with the value 512 (0000 0010 0000 0000 in binary) to enable the event register to be set when limit checking passes (comes back into limit) on channel one, measurement one.

:STATus
:QUEStionable
:NTRansition?

Parameters: none

Response: <NR1>

negative transition mask

Returned values: negative transition mask: integer. Values are in the range 0 to 32767.

Description: Read the mask from the Questionable Status Negative Transition Filter Register. This is a sixteen bit register.

Example: `:STAT:QUES:NTR?`

:STATus :QUEStionable :PTRansition

Parameters: <NRf>

positive transition mask

Valid values: positive transition mask: integer. Valid values are 0 to 65535. Values outside range are rejected and an error generated.

Description: Sets the positive transition filter which allows transitions from 0 to 1 in the Questionable Status Condition Register to be latched into the Questionable Status Event Register.

Bit 15 of the mask value supplied is ignored since bit 15 of the Questionable Status Positive Transition Filter Register is always zero. This is a sixteen bit register.

See Appendix A for more details

Example: `:STAT:QUES:PTR 32`

Program the positive transition filter associated with the Questionable Status Register with the value 512 (0000 0010 0000 0000 in binary) to enable the event register to be set when limit checking fails on channel one, measurement one.

:STATus :QUEStionable :PTRansition?

Parameters: none

Response: <NR1>

positive transition mask

Returned values: positive transition mask: integer. Values are in the range 0 to 32767.

Description: Read the mask from the Questionable Status Positive Transition Filter Register. This is a sixteen bit register.

Example: `:STAT:QUES:PTR?`

STEP subsystem

```
STEP
        DISTance\?
        FREQuency
                AUTO\?
                [VALue]\?
        POWer
                DB\?
                WATTs\?
        TIME
                DELay\?
                [TIME]\?
        UNITs\?
        VOLTage\?
```

:STEP
:DISTance

Parameters: <NUMERIC VALUE>

distance step

Valid values: distance step: real

Suffix: distance step: The suffix that is accepted depends on the domain units:

Domain Units	Allowable suffixes	Default suffix (if none entered)
Distance (m)	mm, m, km	m
Distance (ft)	ft	ft

Description: Set the distance step. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The distance step set is only used when the parameter being altered is a distance.

The distance step applies instrument wide.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example:
```
:STEP:DIST 5M
```
Set distance step to 5 m.

:STEP
:DISTance?

Parameters: none

Response: <NR2>

distance step

Returned values: distance step: real

Description: Determine the distance step. Always returned in units of metres or feet depending of the setting of FLOCation:UNITs.

Example:
```
:FLOC:UNIT FEET;:STEP:DIST 1KM;DIST?
```
Set distance step to 1km and read it back in units of feet. Note that the FLOC:UNIT command will also get the instrument to display all distance data on the screen in units of feet.

Example response: 328.08399

:STEP
:FREQuency
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Set whether the user or the instrument sets the frequency step for the active channel. If the instrument is automatically setting the frequency step then changing the span on the active channel will alter the frequency step so that it is always 0.1 times the span.

This is set 'on' at preset for both channels.

Example: `:STEP:FREQ:AUTO ON`

The instrument sets the frequency step.

:STEP
:FREQuency
:AUTO?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether the user or the instrument sets the frequency step for the active channel.

Example: `:STEP:FREQ:AUTO?`

:STEP

:FREQuency
[:VALue]

Parameters: <NUMERIC VALUE>

frequency step

Valid values: frequency step: real

Suffix: frequency step: The following suffixes are accepted for frequency: Hz, kHz, MHz and GHz. If no suffix is entered then the default suffix of Hz is assumed.

Description: Set the frequency step. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The frequency step set is only used when the parameter being altered is a frequency.

Setting an explicit frequency step automatically uncouples the frequency step from the span, see :STEP:FREQuency:AUTO.

The frequency step applies to the active channel.

See Appendix B for the special values that may be entered instead of a numeric value for NUMERIC VALUE.

Example: `:STEP:FREQ 5E6`

Set frequency step to 5 MHz.

:STEP

:FREQuency
[:VALue]?

Parameters: none

Response: <NR2>

frequency step

Returned values: frequency step: real

Description: Determine the frequency step for the active channel. This will always return the frequency step regardless of whether it was set by the user or the instrument.

Example: `:STEP:FREQ?`

:STEP
:POWer
:DB

Parameters: <NUMERIC VALUE>

db step

Valid values: db step: real

Suffix: db step: A suffix of dB is accepted for power. If no suffix is entered then the default suffix of dB is assumed.

Description: Set the dB step. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The dB step set is only used when the parameter being altered is power and displayed as dB.

The power step applies instrument wide.

Example: `:STEP:POW:DB 2`

Set db step to 2 dB.

:STEP
:POWer
:DB?

Parameters: none

Response: <NR2>

db step

Returned values: db step: real

Description: Determine the dB step.

Example: `:STEP:POW:DB?`

:STEP
:POWer
:WATTs

Parameters: <NUMERIC VALUE>

watts step

Valid values: watts step: real

Suffix: watts step: The following suffixes are accepted for power: mW, W and kW. If no suffix is entered then the default suffix of W is assumed.

Description: Set the watts step. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The watts step set is only used when the parameter being altered is power and displayed as watts.

The power step applies instrument wide.

Example: `:STEP:POW:WATT 0.1`

Set watts step to 0.1 watts.

:STEP
:POWer
:WATTs?

Parameters: none

Response: <NR2>

watts step

Returned values: watts step: real

Description: Determine the watts step.

Example: `:STEP:POW:WATT?`

:STEP
:TIME
:DELay

Parameters: <NUMERIC VALUE>

delay time step

Valid values: delay time step: real

Suffix: delay time step: The following suffixes are accepted for time: ps, ns, us, ms and s. If no suffix is entered then the default suffix of s is assumed.

Description: Set the time step for group delay in seconds. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The time step set is only used when the parameter being altered is a time (group delay).

The time step applies instrument wide.

Example: `:STEP:TIME:DEL 50E-6`

Set time step (group delay) to 50 μs.

:STEP
:TIME
:DELay?

Parameters: none

Response: <NR3>

delay time step

Returned values: delay time step: real

Description: Determine the time step used for group delay time increment/decrement.

Example: `:STEP:TIME:DEL?`

:STEP
:TIME
[:TIME]

Parameters: <NUMERIC VALUE>

time step

Valid values: time step: real

Suffix: time step: The following suffixes are accepted for time: ms and s. If no suffix is entered then the default suffix of s is assumed.

Description: Set the time step in seconds. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The time step set is only used when the parameter being altered is a time.

The time step applies instrument wide.

Example: `:STEP:TIME 50E-3`

Set time step to 50 ms.

:STEP
:TIME
[:TIME]?

Parameters: none

Response: <NR3>

time step

Returned values: time step: real

Description: Determine the time step.

Example: `:STEP:TIME?`

:STEP
:UNITs

Parameters: <NRf>

units step

Valid values: units step: real

Description: Set the units step. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The units step set is only used when the parameter being altered is unitless.

The units step applies instrument wide.

Example: `:STEP:UNIT 4`

Set units step to 4.

:STEP
:UNITs?

Parameters: none

Response: <NR2>

units step

Returned values: units step: real

Description: Determine the units step.

Example:
```
:STEP:UNIT?
```

:STEP
:VOLTage

Parameters: <NUMERIC VALUE>

voltage step

Valid values: voltage step: real

Suffix: voltage step: The following suffixes are accepted for voltage: mV, V and kV. If no suffix is entered then the default suffix of V is assumed.

Description: Set the voltage step. This is the amount that is added/subtracted from the current value when UP or DOWN is sent as a parameter instead of a numeric value. The voltage step set is only used when the parameter being altered is a voltage.

The voltage step applies instrument wide.

Example:
```
:STEP:VOLT 1
```

Set voltage step to 1 V.

:STEP
:VOLTage?

Parameters: none

Response: <NR2>

voltage step

Returned values: voltage step: real

Description: Determine the voltage step.

Example:
```
:STEP:VOLT?
```

SYSTem subsystem

```
SYSTem
    ADDRess
        [SELF]\?
        SOURce\?
    APPLication
        ARUN
            [SELect]\?
            STATe\?
        DISK
            INSTall
            LIST?
            NUMBer?
        LIST?
        NUMBer?
        REMove
        [RUN]
    CONTroller\?
    DATE\?
    DIAGnostics
        DISPlay
    ERRor?
    FREQuency
        STANdard\?
    HOURs?
    IPRights?
    ISETtings
        COUNtry
            CURRent?
            DISK
                INSTall
                LIST?
                NUMBer?
                VERSion?
            LIST?
            NUMBer?
            REMove
            [SELect]
            VERSion?
        DATE\?
        DPOint\?
        KEYBoard
            CURRent?
            DISK
                INSTall
                LIST?
                NUMBer?
                VERSion?
```

```
                LIST?
                NUMBer?
                REMove
                [SELect]
                VERSion?
        LANGuage
                CURRent?
                DISK
                        INSTall
                        LIST?
                        NUMBer?
                        VERSion?
                LIST?
                NUMBer?
                REMove
                [SELect]
                VERSion
                        MINimum?
                        [NUMBer]?
        SEParator\?
        TIME\?
MODE\?
OPTions?
PRESet
SECRet\?
SERial
        BAUD\?
        BITS\?
        FCONtrol\?
        PARity\?
        SBITs\?
SETTings
        ID\?
        RECall
        SAVE
TIME\?
TRIGger
        [MODE]\?
```

:SYSTem :ADDRess [:SELF]

Parameters: <NRf>

address

Valid values: address: integer. Valid values are 0 to 30. Values outside range are rejected and an error generated.

Description: Set the GPIB address of the instrument.

It is recommended that a *OPC? is executed prior to this command (and the resulting 1 read by the controller) and a delay is inserted after using this command to give the instrument time to execute it.

Example: `:SYST:ADDR 18`

Set instrument address to 18.

:SYSTem :ADDRess [:SELF]?

Parameters: none

Response: <NR1>

address

Returned values: address: integer. Values are in the range 0 to 30.

Description: Determine the GPIB address of the instrument.

Example: `:SYST:ADDR?`

:SYSTem :ADDRess :SOURce

Parameters: <NRf>

address

Valid values: address: integer. Valid values are 0 to 30. Values outside range are rejected and an error generated.

Description: Set the GPIB address of the external source that the instrument controls during some calibrations.

Example: `:SYST:ADDR:SOUR 9`

Set external source address to 9.

:SYSTem :ADDRess :SOURce?

Parameters: none

Response: <NR1>

address

Returned values: address: integer. Values are in the range 0 to 30.

Description: Determine the GPIB address of the external source.

Example:
```
:SYST:ADDR:SOUR?
```

:SYSTem :APPLication :ARUN [:SELect]

Parameters: <STRING PROGRAM DATA>

application

Valid values: application: string. Maximum length of 256 characters excluding quotes. Excess characters will be ignored.

Description: Selects the application that should be auto-run at power up and on preset.. The application string must be one of the strings read using :SYST:APPL:LIST? or an error will be generated. Selecting an application to be autorun will automatically turn on application auto-run.

Example:
```
:SYST:APPL:ARUN "INSERTION LOSS"
```

Select insertion loss application to autorun at power up or preset.

:SYSTem :APPLication :ARUN [:SELect]?

Parameters: none

Response: <STRING RESPONSE DATA>

application

Returned values: application: string. Maximum length of 256 characters excluding quotes.

Description: Determine which application is set up to be autorun. The name will be returned event if the auto-run state is off..

Example:
```
:SYST:APPL:ARUN?
```

Determine which application will be autorun (if autorun is enabled).

:SYSTem :APPLication :ARUN :STATe

Parameters: <BOOLEAN PROGRAM DATA>

state

Description: Enable or disable auto-run of application. If enabled, the application specified using :SYST:APPL:ARUN is automatically run at power up and preset..

Example:
```
:SYST:APPL:ARUN:STAT ON
```

Enable auto-run of application.

:SYSTem :APPLication :ARUN :STATe?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

state

Description: Determine whether an application is set up to auto-run or not.

Example:
```
:SYST:APPL:ARUN:STAT?
```

:SYSTem :APPLication :DISK :INSTall

Parameters: <STRING PROGRAM DATA>

application

Valid values: application: string. Maximum length of 256 characters excluding quotes. Excess characters will be ignored.

Description: Install a new application into the instrument from the floppy disk. The application string must be one of the strings read using :SYST:APPL:DISK:LIST? or an error will be generated. Once the new application has been installed into the instrument it can be run using :SYST:APPL (it is not automatically run).

Example:
```
:SYST:APPL:DISK:INST "6800 DEMO"
```

Install demo application into the instrument.

:SYSTem
:APPLication
:DISK
:LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first application, ..., last application

Returned values: first application: string. Maximum length of 256 characters excluding quotes.
...
last application: string. Maximum length of 256 characters excluding quotes.

Description: List all applications on the floppy disk that are available to be installed into the instrument. The disk containing the applications(s) to be installed must be present. To install one of the applications listed, use :SYST:APPL:DISK:INST using one of the strings returned by this command.

Example:
```
:SYST:APPL:DISK:LIST?
```
List all the applications on the floppy disk

:SYSTem
:APPLication
:DISK
:NUMBer?

Parameters: none

Response: <NR1>

number of applications

Returned values: number of applications: integer

Description: This command is to be used in conjunction with :SYST:APPL:DISK:LIST. This command determines how many applications will be returned by the :SYST:APPL:DISK:LIST? command. This can be useful in reserving space for the :SYST:APPL:DISK:LIST? response.

Example:
```
:SYST:APPL:DISK:NUMB?
```
Determine how many applications are on the floppy disk

:SYSTem :APPLication :LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first application, ..., last application

Returned values: first application: string. Maximum length of 256 characters excluding quotes.
...
last application: string. Maximum length of 256 characters excluding quotes.

Description: List all applications available in the instrument. An application may then be selected using :SYST:APPL with one of the strings returned by this command.

Example: `:SYST:APPL:LIST?`

List all the applications available.

:SYSTem :APPLication :NUMBer?

Parameters: none

Response: <NR1>

number of applications

Returned values: number of applications: integer

Description: This command is to be used in conjunction with :SYST:APPL:LIST. This command determines how many applications will be returned by the :SYST:APPL:LIST? command. This can be useful in reserving space for the :SYST:APPL:LIST? response.

Example: `:SYST:APPL:NUMB?`

Determine how many applications are available in the instrument.

:SYSTem
:APPLication
:REMove

Parameters: <STRING PROGRAM DATA>

application

Valid values: application: string. Maximum length of 256 characters excluding quotes. Excess characters will be ignored.

Description: Remove an application from the instrument. The application string must be one of the strings read using :SYST:APPL:LIST? or an error will be generated. It may be useful to remove applications from the instrument if it is known that they will not be used since it will allow memory to be freed for storage of other items.

Applications that are built into the instrument (i.e. not installed from floppy) cannot be removed.

Example: `:SYST:APPL:REM "6800 Demo"`

Remove 6800 Demo application from the instrument.

:SYSTem
:APPLication
[:RUN]

Parameters: <STRING PROGRAM DATA>

application

Valid values: application: string. Maximum length of 256 characters excluding quotes. Excess characters will be ignored.

Description: Load and run an application. The application string must be one of the strings read using :SYST:APPL:LIST? or an error will be generated.

Example: `:SYST:APPL "INSERTION LOSS"`

Run insertion loss application.

:SYSTem :CONTroller

Parameters: <CPD>

controller

Valid values: controller: [SERial | GPIB | NONE]. Values other than those stated are rejected and an error generated.

Description: This command sets how the instrument is being controlled. It can be controlled by either a controller on the GPIB bus or by a controller over RS-232. It is also possible to select no controller.

If no controller is selected then hardcopy outputs can be performed to both RS-232 and GPIB by the MMI. The instrument will be a system controller on the GPIB bus.

If GPIB controller is selected then the instrument will accept remote commands from the GPIB bus. The instrument will be able to take control to perform hardcopy output to the GPIB bus and will pass control back at the end of the plot. Hardcopy output to RS-232 will be permitted.

If serial controller is selected then the instrument will accept remote commands from the RS-232. Hardcopy output to RS-232 will not be possible. Hardcopy to GPIB will be permitted and the instrument will set itself up to be system controller.

Example: `:SYST:CONT GPIB`

Select the GPIB bus for use by an external controller.

:SYSTem :CONTroller?

Parameters: none

Response: <CRD>

controller

Returned values: controller: [SER | GPIB | NONE]

Description: Determine whether the instrument is accepting external control and if so, where from..

Example: `:SYST:CONT?`

:SYSTem :DATE

Parameters: <NRf>, <NRf>, <NRf>

year, month, day

Valid values: year: integer

month: integer. Valid values are 1 to 12. Values outside range are rejected and an error generated.

day: integer. Valid values are 1 to 31. Values outside range are rejected and an error generated.

Description: Sets the date of the real-time clock. The year must be entered in full. Invalid dates, e.g. 1995,2,31, will be rejected and an error generated.

Example: `:SYST:DATE 1998, 8, 20`

Set the date to 20th August 1998.

:SYSTem :DATE?

Parameters: none

Response: <NR1>, <NR1>, <NR1>

year, month, day

Returned values: year: integer
month: integer. Values are in the range 1 to 12.
day: integer. Values are in the range 1 to 31.

Description: Read the date of the real-time clock

Example: `:SYST:DATE?`

:SYSTem :DIAGnostics :DISPlay

Parameters: <CPD>

display diagnostics mode

Valid values: display diagnostics mode: [WHITe | BLACk | NORMal]. Values other than those stated are rejected and an error generated.

Description: This command allows the display to be tested for stuck pixels.

Setting the display to WHITe will turn all pixels on - allowing a pixels stuck off test.

Setting the display to BLACk will turn all pixels off - allowing a pixels stuck on test.

Sending NORMal returns the display back to the normal mode.

Sending any other command whilst in display test mode will action the command as normal but the effect on the display is undefined. It is recommended that the only non-query commands sent whilst in display test mode are :SYST:DIAG:DISP NORM, *RST or :SYST:PRES.

Example: `:SYST:DIAG:DISP WHIT`

Turn all the pixels on the display on (to test for any pixels stuck off).

:SYSTem :ERRor?

Parameters: none

Response: <NR1>,<STRING RESPONSE DATA>

error number, error message string

Returned values: error number: integer
error message string: string

Description: Read the SCPI 1990.0 error number and error message from the head of the error queue.

Example: `:SYST:ERR?`

Example Response: `-112,"Program mnemonic too long"`

:SYSTem :FREQuency :STANdard

Parameters: <CPD>

frequency standard

Valid values: frequency standard: [INT | EX1 | EX10]. Values other than those stated are rejected and an error generated.

Description: Select a frequency standard for the instrument: internal, external 1 MHz or external 10 MHz. This command is also valid in source only mode.

This command is equivalent to SOUR:FREQ:STAN.

Example: `:SYST:FREQ:STAN EX1`

Select the rear panel FREQ STD INPUT/OUTPUT to accept a 1 MHz external frequency standard.

:SYSTem :FREQuency :STANdard?

Parameters: none

Response: <CRD>

frequency standard

Returned values: frequency standard: [INT | EX1 | EX10]

Description: Determine the frequency standard in use. This command is also valid in source only mode.

This command is equivalent to SOUR:FREQ:STAN?.

Example: `:SYST:FREQ:STAN?`

:SYSTem :HOURs?

Parameters: none

Response: <NR1>

operating hours

Returned values: operating hours: integer

Description: Determine how long the instrument has been operating. The value returned is in units of hours, rounded to the nearest hour

Example: `:SYST:HOUR?`

:SYSTem
:IPRights?

Parameters: none

Response: <STRING RESPONSE DATA>

message

Returned values: message: string

Description: Return intellectual property rights message.

Example: `:SYST:IPR?`

Return IPR message.

:SYSTem
:ISETtings
:COUNtry
:CURRent?

Parameters: none

Response: <STRING RESPONSE DATA>

country in use

Returned values: country in use: string. Maximum length of 30 characters excluding quotes.

Description: Determine which county is selected. The instrument will always have at least one country available : UK. Other countries may be built into the instrument and further countries can be installed from floppy disk. Use :SYST:ISET:COUN:LIST? to determine which countries are currently available on the instrument. Use :SYST:ISET:COUN[:SEL] to select a different country.

Example: `:SYST:ISET:COUN:CURR?`

:SYSTem :ISETtings :COUNtry :DISK :INSTall

Parameters: <STRING PROGRAM DATA>

country

Valid values: country: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Install a new country into the instrument along with its associated language and keyboard. The country string must be one of the strings read using :SYST:ISET:COUN:DISK:LIST? or an error will be generated. Once the new country has been installed into the instrument it can be selected using :SYST:ISET:COUN:SEL (it is not automatically selected).

Example:
```
:SYST:ISET:COUN:DISK:INST "FRANCE"
```
Install default settings for France into the instrument.

:SYSTem :ISETtings :COUNtry :DISK :LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first country, ..., last country

Returned values: first country: string. Maximum length of 30 characters excluding quotes.
...
last country: string. Maximum length of 30 characters excluding quotes.

Description: List all countries on the floppy disk that are available to be installed into the instrument. The disk containing the country(s) to be installed must be present. To install one of the countries listed, use :SYST:ISET:COUN:DISK:INST using one of the strings returned by this command.

Example:
```
:SYST:ISET:COUN:DISK:LIST?
```
List all the countries on the floppy disk

:SYSTem :ISETtings :COUNtry :DISK :NUMBer?

Parameters: none

Response: <NR1>

number of countries

Returned values: number of countries: integer

Description: This command is to be used in conjunction with :SYST:ISET:COUN:DISK:LIST. This command determines how many countries will be returned by the :SYST:ISET:COUN:DISK:LIST? command. This can be useful in reserving space for the :SYST:ISET:COUN:DISK:LIST? response.

Example:
```
:SYST:ISET:COUN:DISK:NUMB?
```

Determine how many countries are on the floppy disk

:SYSTem :ISETtings :COUNtry :DISK :VERSion?

Parameters: <STRING PROGRAM DATA>

country

Valid values: country: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

version number

Returned values: version number: integer

Description: Determine the version number of the specified country file on the floppy disk. The country string must be one of the strings read using :SYST:ISET:COUN:DISK:LIST? or an error will be generated.

Example:
```
:SYST:ISET:COUN:DISK:VERS? "FRANCE"
```

Determine version number for France *country file on the floppy.*

:SYSTem
:ISETtings
:COUNtry
:LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first country, ..., last country

Returned values: first country: string. Maximum length of 30 characters excluding quotes.
...
last country: string. Maximum length of 30 characters excluding quotes.

Description: List all countries available in the instrument. A country may then be selected using :SYST:ISET:COUN with one of the strings returned by this command.

Example:
```
:SYST:ISET:COUN:LIST?
```
List all the countries available.

:SYSTem
:ISETtings
:COUNtry
:NUMBer?

Parameters: none

Response: <NR1>

number of countries

Returned values: number of countries: integer

Description: This command is to be used in conjunction with :SYST:ISET:COUN:LIST. This command determines how many countries will be returned by the :SYST:ISET:COUN:LIST? command. This can be useful in reserving space for the :SYST:ISET:COUN:LIST? response.

Example:
```
:SYST:ISET:COUN:NUMB?
```
Determine how many countries are available in the instrument.

:SYSTem
:ISETtings
:COUNtry
:REMove

Parameters: <STRING PROGRAM DATA>

country

Valid values: country: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Remove a county from the instrument.

Example: `:SYST:ISET:COUN:REM "FRANCE"`

Remove France country settings from the instrument.

:SYSTem
:ISETtings
:COUNtry
[:SELect]

Parameters: <STRING PROGRAM DATA>

country

Valid values: country: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Select a county. This will default all internationalisation parameters to those normally associated with that country. The country string must be one of the strings read using :SYST:ISET:COUN:LIST? or an error will be generated.

Selecting a new country is equivalent to obeying the following commands

SYST:ISET:DATE
SYST:ISET:DPO
SYST:ISET:KEYB
SYST:ISET:LANG
SYST:ISET:SEP
SYST:ISET:TIME

with the default values for that country.

Example: `:SYST:ISET:COUN "UK"`

Select UK defaults.

:SYSTem
:ISETtings
:COUNtry
:VERSion?

Parameters: <STRING PROGRAM DATA>

country

Valid values: country: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

version number

Returned values: version number: integer

Description: Determine the version number of the specified country file installed in the instrument. The country string must be one of the strings read using :SYST:ISET:COUN:LIST? or an error will be generated.

Example: `:SYST:ISET:COUN:VERS? "FRANCE"`

Determine version number for France *country file installed on the instrument.*

:SYSTem
:ISETtings
:DATE

Parameters: <CPD>, <CPD>

date order, date separator

Valid values: date order: [MDY | YMD | DMY]. Values other than those stated are rejected and an error generated.
date separator: [SLASh | MINus | DOT]. Values other than those stated are rejected and an error generated.

Description: Select the order for the date output. The following are accepted:

MDY	gives format 12 18 1998
YMD	gives format 1998 12 18
DMY	gives format 18 12 1998

The separator character is placed between the 3 numbers that are output. The following are accepted:

SLASh	gives format 28/02/1998
MINus	gives format 28-02-1998
DOT	gives format 28.02.1998

Leading zeros are always displayed giving two digits for both the day and the month.

Example: `:SYST:ISET:DATE DMY,SLAS`

Select the 'UK format' using slashes to give e.g. 21/04/1998.

:SYSTem :ISETtings :DATE?

Parameters: none

Response: <CRD>, <CRD>

date order, date separator

Returned values: date order: [MDY | YMD | DMY]
date separator: [SLAS | MIN | DOT]

Description: Determine the format that will be used to output dates.

Example:
```
:SYST:ISET:DATE?
```

:SYSTem :ISETtings :DPOint

Parameters: <CPD>

decimal point character

Valid values: decimal point character: [DOT | COMMa]. Values other than those stated are rejected and an error generated.

Description: Sets which character ('.' or ',') is used to separate the integral and fractional part of real numbers. Note that this does not alter how real numbers are output to a remote controller.

If the setting is changed to comma and the spreadsheet field separator (as set by :SYST:ISET:SEP) is also set to comma then the spreadsheet field separator will be forced to semicolon.

Example:
```
:SYST:ISET:DPO DOT
```

Display numbers and output numbers to CSV memories as xx.yyy.

:SYSTem :ISETtings :DPOint?

Parameters: none

Response: <CRD>

decimal point character

Returned values: decimal point character: [DOT | COMM]

Description: Determine whether numbers are displayed with a '.' or a ',' for the decimal point.

Example:
```
:SYST:ISET:DPO?
```

:SYSTem :ISETtings :KEYBoard :CURRent?

Parameters: none

Response: <STRING RESPONSE DATA>

keyboard in use

Returned values: keyboard in use: string. Maximum length of 30 characters excluding quotes.

Description: Determine which keyboard is in use. The instrument will always have at least one keyboard available : UK. Other keyboards may be built into the instrument and further keyboards can be installed from floppy disk. Use :SYST:ISET:KEYB:LIST? to determine which keyboards are currently available on the instrument. Use :SYST:ISET:KEYB[:SEL] to select a different keyboard.

Example:
```
:SYST:ISET:KEYB:CURR?
```

:SYSTem :ISETtings :KEYBoard :DISK :INSTall

Parameters: <STRING PROGRAM DATA>

keyboard

Valid values: keyboard: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Install a new keyboard into the instrument. The keyboard string must be one of the strings read using :SYST:ISET:KEYB:DISK:LIST? or an error will be generated. Once the new keyboard has been installed into the instrument it can be selected using :SYST:ISET:KEYB:SEL (it is not automatically selected).

Example:
```
:SYST:ISET:KEYB:DISK:INST "FRENCH"
```

Install French keyboard driver into the instrument.

:SYSTem :ISETtings :KEYBoard :DISK :LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first keyboard, ..., last keyboard

Returned values: first keyboard: string. Maximum length of 30 characters excluding quotes.
...
last keyboard: string. Maximum length of 30 characters excluding quotes.

Description: List all keyboards on the floppy disk that are available to be installed into the instrument. The disk containing the keyboard(s) to be installed must be present. To install one of the keyboards listed, use :SYST:ISET:KEYB:DISK:INST using one of the strings returned by this command.

Example:
```
:SYST:ISET:KEYB:DISK:LIST?
```
List all the keyboards on the floppy disk

:SYSTem :ISETtings :KEYBoard :DISK :NUMBer?

Parameters: none

Response: <NR1>

number of keyboards

Returned values: number of keyboards: integer

Description: This command is to be used in conjunction with :SYST:ISET:KEYB:DISK:LIST. This command determines how many keyboards will be returned by the :SYST:ISET:KEYB:DISK:LIST? command. This can be useful in reserving space for the :SYST:ISET:KEYB:DISK:LIST? response.

Example:
```
:SYST:ISET:KEYB:DISK:NUMB?
```
Determine how many keyboards are on the floppy disk

:SYSTem
:ISETtings
:KEYBoard
:DISK
:VERSion?

Parameters: <STRING PROGRAM DATA>

keyboard

Valid values: keyboard: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

version number

Returned values: version number: integer

Description: Determine the version number of the specified keyboard file on the floppy disk. The keyboard string must be one of the strings read using :SYST:ISET:KEYB:DISK:LIST? or an error will be generated.

Example: `:SYST:ISET:KEYB:DISK:VERS? "FRENCH"`

Determine version number for French *keyboard file on the floppy.*

:SYSTem
:ISETtings
:KEYBoard
:LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first keyboard, ..., last keyboard

Returned values: first keyboard: string. Maximum length of 30 characters excluding quotes.
...
last keyboard: string. Maximum length of 30 characters excluding quotes.

Description: List all keyboards available in the instrument. A keyboard may then be selected using :SYST:ISET:KEYB with one of the strings returned by this command.

Example: `:SYST:ISET:KEYB:LIST?`

List all the keyboards available.

:SYSTem :ISETtings :KEYBoard :NUMBer?

Parameters: none

Response: <NR1>

number of keyboards

Returned values: number of keyboards: integer

Description: This command is to be used in conjunction with :SYST:ISET:KEYB:LIST. This command determines how many keyboards will be returned by the :SYST:ISET:KEYB:LIST? command. This can be useful in reserving space for the :SYST:ISET:KEYB:LIST? response.

Example:
```
:SYST:ISET:KEYB:NUMB?
```
Determine how many keyboards are available in the instrument.

:SYSTem :ISETtings :KEYBoard :REMove

Parameters: <STRING PROGRAM DATA>

keyboard

Valid values: keyboard: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Remove a keyboard driver from the instrument.

Example:
```
:SYST:ISET:KEYB:REM "GERMAN"
```
Remove German keyboard driver.

:SYSTem
:ISETtings
:KEYBoard
[:SELect]

Parameters: <STRING PROGRAM DATA>

keyboard

Valid values: keyboard: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Select a keyboard. The keyboard string must be one of the strings read using :SYST:ISET:KEYB:LIST? or an error will be generated.

To install new keyboard layouts onto the instrument it is necessary to use the :SYST:ISET:COUN:DISK:INST command to install all the data for a particular country. For instance to install the keyboard for a Spanish keyboard, install the data associated with the country Spain and then use this command to select the Spanish keyboard that will now be present on the instrument.

Example: `:SYST:ISET:KEYB "UK"`

Select UK keyboard.

:SYSTem
:ISETtings
:KEYBoard
:VERSion?

Parameters: <STRING PROGRAM DATA>

keyboard

Valid values: keyboard: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>

version number

Returned values: version number: integer

Description: Determine the version number of the specified keyboard file installed in the instrument. The keyboard string must be one of the strings read using :SYST:ISET:KEYB:LIST? or an error will be generated.

Example: `:SYST:ISET:KEYB:VERS? "FRENCH"`

Determine version number for French *keyboard driver installed on the instrument.*

:SYSTem :ISETtings :LANGuage :CURRent?

Parameters: none

Response: <STRING RESPONSE DATA>

language in use

Returned values: language in use: string. Maximum length of 30 characters excluding quotes.

Description: Determine which language is in use. The instrument will always have at least one language available : ENGLISH. Other languages may be built into the instrument and further languages can be installed from floppy disk. Use :SYST:ISET:LANG:LIST? to determine which languages are currently available on the instrument. Use :SYST:ISET:LANG[:SEL] to select a different language.

Example:
```
:SYST:ISET:LANG:CURR?
```

:SYSTem :ISETtings :LANGuage :DISK :INSTall

Parameters: <STRING PROGRAM DATA>

language

Valid values: language: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Install a new language into the instrument. The language string must be one of the strings read using :SYST:ISET:LANG:DISK:LIST? or an error will be generated. Once the new language has been installed into the instrument it can be selected using :SYST:ISET:LANG:SEL (it is not automatically selected).

Example:
```
:SYST:ISET:LANG:DISK:INST "FRENCH"
```

Install French language into the instrument.

:SYSTem
:ISETtings
:LANGuage
:DISK
:LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first language, ..., last language

Returned values: first language: string. Maximum length of 30 characters excluding quotes.
...
last language: string. Maximum length of 30 characters excluding quotes.

Description: List all languages on the floppy disk that are available to be installed into the instrument. The disk containing the language(s) to be installed must be present. To install one of the languages listed, use :SYST:ISET:LANG:DISK:INST using one of the strings returned by this command.

Example: `:SYST:ISET:LANG:DISK:LIST?`

List all the languages on the floppy disk

:SYSTem
:ISETtings
:LANGuage
:DISK
:NUMBer?

Parameters: none

Response: <NR1>

number of languages

Returned values: number of languages: integer

Description: This command is to be used in conjunction with :SYST:ISET:LANG:DISK:LIST. This command determines how many languages will be returned by the :SYST:ISET:LANG:DISK:LIST? command. This can be useful in reserving space for the :SYST:ISET:LANG:DISK:LIST? response.

Example: `:SYST:ISET:LANG:DISK:NUMB?`

Determine how many languages are on the floppy disk

:SYSTem
:ISETtings
:LANGuage
:DISK
:VERSion?

Parameters: <STRING PROGRAM DATA>

language

Valid values: language: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>, <NR1>

version number of display strings, version number of error strings

Returned values: version number of display strings: integer

version number of error strings: integer

Description: Determine the version number of the specified language file on the floppy disk. The language string must be one of the strings read using :SYST:ISET:LANG:DISK:LIST? or an error will be generated.

Example:
```
:SYST:ISET:LANG:DISK:VERS? "FRENCH"
```
Determine version number for French *language file on the floppy.*

:SYSTem :ISETtings :LANGuage :LIST?

Parameters: none

Response: <STRING RESPONSE DATA>, ..., <STRING RESPONSE DATA>

first language, ..., last language

Returned values: first language: string. Maximum length of 30 characters excluding quotes.
...
last language: string. Maximum length of 30 characters excluding quotes.

Description: List all languages available in the instrument. A language may then be selected using :SYST:ISET:LANG with one of the strings returned by this command.

Example:
```
:SYST:ISET:LANG:LIST?
```
List all the languages available.

:SYSTem :ISETtings :LANGuage :NUMBer?

Parameters: none

Response: <NR1>

number of languages

Returned values: number of languages: integer

Description: This command is to be used in conjunction with :SYST:ISET:LANG:LIST. This command determines how many languages will be returned by the :SYST:ISET:LANG:LIST? command. This can be useful in reserving space for the :SYST:ISET:LANG:LIST? response.

Example:
```
:SYST:ISET:LANG:NUMB?
```
Determine how many languages are available in the instrument.

:SYSTem
:ISETtings
:LANGuage
:REMove

Parameters: <STRING PROGRAM DATA>

language

Valid values: language: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Remove specified language support from the instrument.

Example:
```
:SYST:ISET:LANG:REM "SPANISH"
```

Remove Spanish language support.

:SYSTem
:ISETtings
:LANGuage
[:SELect]

Parameters: <STRING PROGRAM DATA>

language

Valid values: language: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Description: Select a language for displayed text. The language string must be one of the strings read using :SYST:ISET:LANG:LIST? or an error will be generated.

Selecting a new language will affect text displayed on the screen and the text in the GPIB error messages.

Example:
```
:SYST:ISET:LANG "ENGLISH"
```

Select English language.

:SYSTem :ISETtings :LANGuage :VERSion :MINimum?

Parameters: none

Response: <NR1>, <NR1>

version number of display strings, version number of error strings

Returned values: version number of display strings: integer

version number of error strings: integer

Description: Determine the minimum language version numbers that can be used with the instrument. Language files with older version numbers than this cannot be used with the instrument.

Example: `:SYST:ISET:LANG:VERS:MIN?`

Determine minimum version numbers for language files that may be used with the instrument.

:SYSTem :ISETtings :LANGuage :VERSion [:NUMBer]?

Parameters: <STRING PROGRAM DATA>

language

Valid values: language: string. Maximum length of 30 characters excluding quotes. Excess characters will be ignored.

Response: <NR1>, <NR1>

version number of display strings, version number of error strings

Returned values: version number of display strings: integer

version number of error strings: integer

Description: Determine the version number of the specified language file installed in the instrument. The language string must be one of the strings read using :SYST:ISET:LANG:LIST? or an error will be generated.

Example: `:SYST:ISET:LANG:VERS? "FRENCH"`

Determine version number for French *language installed on the instrument.*

:SYSTem :ISETtings :SEParator

Parameters: <CPD>

separator

Valid values: separator: [SEMicolon | COMMa]. Values other than those stated are rejected and an error generated.

Description: Sets which character (';' or ',') is used to separate the values output to CSV format memories.

If the setting is changed to comma and the decimal point character (as set by :SYST:ISET:DPO) is also set to comma then the decimal point character will be forced to dot.

Example: `:SYST:ISET:SEP SEM`

Separate values saved to CSV memories with a semicolon.

:SYSTem :ISETtings :SEParator?

Parameters: none

Response: <CRD>

separator

Returned values: separator: [SEM | COMM]

Description: Determine whether the values output to a CSV memory are separated with a semicolon or a comma.

Example: `:SYST:ISET:SEP?`

:SYSTem :ISETtings :TIME

Parameters: <BOOLEAN PROGRAM DATA>, <CPD>

24 hour format, time separator

Valid values: time separator: [COLon | COMMa | DOT]. Values other than those stated are rejected and an error generated.

Description: Selects whether times are output as twelve (OFF) or twenty four (ON) hour format and which character is placed between the 3 numbers that are output. The following are accepted:

COLon	gives format 22:02:19 or 10:02:19p
COMMa	gives format 22,02,19 or 10,02,19p
DOT	gives format 22.02.19 or 10.02.19p

Leading zeros are always displayed except for hours displayed in 12-hour format.

Example:
```
:SYST:ISET:TIME ON,COL
```
Select the 'UK format' using colons to give e.g. 13:04:59.

:SYSTem :ISETtings :TIME?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>, <CRD>

24 hour format, time separator

Returned values: time separator: [COL | COMM | DOT]

Description: Determine the format that will be used to output times.

Example:
```
:SYST:ISET:TIME?
```

:SYSTem :MODE

Parameters: <CPD>

instrument mode

Valid values: instrument mode: [MEASurement | SOURce]. Values other than those stated are rejected and an error generated.

Description: Selects whether the instrument is in its normal operating mode (measuring) or in source only mode:

MEASurement	instrument in normal operation
SOURce	instrument in source only mode

Example:
```
:SYST:MODE SOUR
```
Select source only mode.

:SYSTem :MODE?

Parameters: none

Response: <CRD>

instrument mode

Returned values: instrument mode: [MEAS | SOUR]

Description: Determine whether the instrument is in source only mode or not.

Example: `:SYST:MODE?`

:SYSTem :OPTions?

Parameters: none

Response: <NR1>

options

Returned values: options: integer. Values are in the range 0 to 32767.

The returned value is a sixteen bit value *with each set bit representing the presence of an option:*

Bit Number	Bit Value	Option Present If Bit Set
0	1	Monochrome display
1	2	70 dB Attenuator (46 GHz)
2	4	70 dB Attenuator (20 GHz)
3	8	90 dB Attenuator (24 GHz)
4	16	110 dB Attenuator (3 GHz)
5	32	132 dB Attenuator (3 GHz)
6	64	AM & Wideband FM Demodulator
7	128	External mixer
8	256	Field replaceable connectors
9	512	Group Delay
10	1024	Source Frequency Modulation
11	2048	High Power
12	4096	Source FM (external modulation only)
13	8192	SPARE
14	16384	SPARE
15	32768	Always zero

This command can be used instead of *OPT? if it is easier to decode than the text strings returned by *OPT?.

Bits 10 and 12 are mutually exclusive. Bit 10 will be set if the internal FM option is present (and therefore both internal and external FM is available) otherwise bit 12 will be set to indicate that external FM only is present.

Description: Read hardware options present.

Example: `:SYST:OPT?`

:SYSTem :PRESet

Parameters: none

Description: Preset measurements on the active channel. If channel coupling is enabled and the non-active channel is the same type as the active channel, then the measurements in the non-active channel will also be preset.

The channel type is not altered.

To place the instrument in its default state, use *RST. See Appendix C for details on which parameters are not affected by *RST or SYST:PRES.

See also *RST in Common Commands. Note that this command is not identical to *RST.

Example: `:SYST:PRES`

Preset the instrument.

:SYSTem :SECRet

Parameters: <BOOLEAN PROGRAM DATA>

secret frequency display

Description: When selected, all X-axis frequency information will be removed from the display and hardcopy.

Does not affect frequency values returned to remote controllers.

Example: `:SYST:SECR ON`

Stop displaying frequency information.

:SYSTem :SECRet?

Parameters: none

Response: <BOOLEAN RESPONSE DATA>

secret frequency display

Description: Determine whether frequency information is being displayed or not.

Example: `:SYST:SECR?`

:SYSTem :SERial :BAUD

Parameters: <NRf>

baud rate

Valid values: baud rate: integer. Valid values are 1200 to 115200. Values outside range are clipped.

Description: Set the serial interface baud rate. The same rate is used for transmission and reception of data. The following values are valid and the nearest will be used:
1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

If this command is being sent over the RS232 port then it is recommended that it is the last command in the program message. A delay should be inserted before sending any further commands at the new baud rate.

Example: `:SYST:SER:BAUD 19200`

Set RS232 serial interface baud rate to 19200.

:SYSTem :SERial :BAUD?

Parameters: none

Response: <NR1>

baud rate

Returned values: baud rate: integer

Description: Determine the serial interface baud rate.

Example: `:SYST:SER:BAUD?`

:SYSTem :SERial :BITS

Parameters: <NRf>

data bits

Valid values: data bits: integer. Valid values are 7 to 8. Values outside range are clipped.

Description: Set the number of data bits sent/received over the RS232 serial bus.

The usual setting for most printers/plotters is 8 data bits, no parity and 1 stop bit.

Example: `:SYST:SER:BITS 8`

Set the number of data bits transmitted per byte over the RS232 to 8 bits.

:SYSTem :SERial :BITS?

Parameters: none

Response: <NR1>

data bits

Returned values: data bits: integer. Values are in the range 7 to 8.

Description: Determine the number of data bits used for transmissions over the serial interface.

Example: `:SYST:SER:BITS?`

:SYSTem
:SERial
:FCONtrol

Parameters: <CPD>

flow control method

Valid values: flow control method: [NONE | XON | HARDware]. Values other than those stated are rejected and an error generated.

Description: Set the flow control method for the RS232 serial port:

NONE	No flow control method in use. Data can be lost if the receiving device is slower than the transmitting device.
XON	Use software handshaking (XON and XOFF).
HARDware	Use hardware handshaking (RTS and CTS).

Note that to use hardware handshaking, the cable in use must contain the correct wires to support the hardware handshaking method.
Both handshaking methods will only work if both devices connected are using the specified method.

Example: `:SYST:SER:FCON HARD`

Set RS232 serial interface to use hardware handshaking.

:SYSTem
:SERial
:FCONtrol?

Parameters: none

Response: <CRD>

flow control method

Returned values: flow control method: [NONE | XON | HARD]

Description: Determine the serial interface flow control method.

Example: `:SYST:SER:FCON?`

:SYSTem :SERial :PARity

Parameters: <CPD>

parity

Valid values: parity: [NONE | EVEN | ODD]. Values other than those stated are rejected and an error generated.

Description: Set the parity for the RS232 serial interface:

NONE	No parity checking in use.
EVEN	Use even parity checking.
ODD	Use odd parity checking.

The usual setting for most printers/plotters is 8 data bits, no parity and 1 stop bit.

Example: `:SYST:SER:PAR NONE`

Set parity checking off.

:SYSTem :SERial :PARity?

Parameters: none

Response: <CRD>

parity

Returned values: parity: [NONE | EVEN | ODD]

Description: Determine the serial interface parity checking.

Example: `:SYST:SER:PAR?`

:SYSTem :SERial :SBITs

Parameters: <NRf>

stop bits

Valid values: stop bits: integer. Valid values are 1 to 2. Values outside range are clipped.

Description: Set the number of stop bits sent/received over the RS232 serial bus.

The usual setting for most printers/plotters is 8 data bits, no parity and 1 stop bit.

Example: `:SYST:SER:SBIT 1`

Set the number of stop bits to 1.

:SYSTem :SERial :SBITs?

Parameters: none

Response: <NR1>

stop bits

Returned values: stop bits: integer. Values are in the range 1 to 2.

Description: Determine the number of stop bits used for transmissions over the serial interface.

Example: `:SYST:SER:SBIT?`

:SYSTem :SETTings :ID

Parameters: <STRING PROGRAM DATA>, <STRING PROGRAM DATA>

file name, user id

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

user id: string

Description: Set the user entered id string of the specified settings store.

Example: `:SYST:SETT:ID "john02", "My favourite settings"`

:SYSTem :SETTings :ID?

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Response: <STRING RESPONSE DATA>

id string

Returned values: id string: string

Description: Read the user entered id string of the specified source power calibration. The id string gives additional details of the power calibration store over what can be determined from the file name alone.

Example: `:SYST:SETT:ID? "mysett"`

:SYSTem
:SETTings
:RECall

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Recall instrument settings from a store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if reading from floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

If it is necessary to set up the instrument from saved settings (downloaded to the controller previously) then this can be done by first loading the data back to a specified settings store and then recalling that store:

```
:MMEM:MSIS "C"
:MMEM:WRIT:SETT "temp", #..lots of binary data...
;SYST:SETT:REC "temp"
:MMEM:DEL "temp.set"
```

Example: `:SYST:SETT:REC "IFR_11"`

Recall settings from store IFR_11.

:SYSTem
:SETTings
:SAVE

Parameters: <STRING PROGRAM DATA>

file name

Valid values: file name: string. Maximum length of 8 characters excluding quotes. Excess characters will be ignored.

Description: Save current instrument settings to a store. The store will be accessed using the file name sent as part of this command, together with the currently selected storage device (internal memory or floppy disk) and the current directory (if writing to floppy disk). A file extension should not be specified as this is fixed by the instrument.

Use :MMEM:MSIS to select instrument memory or floppy disk. Use :MMEM:CDIR to select the required directory when accessing the floppy disk.

If it is necessary to transfer the current settings to the controller, this can be done by saving to a specified store and then transferring that store to the controller. The store can then be deleted:

```
MMEM:MSIS "C"
:SYST:SETT:SAVE "temp"
:MMEM:READ:SETT? "temp"
:MMEM:DEL "temp.set"
```

Example:
```
:SYST:SETT:SAVE "IFR_5"
```

Save instrument settings to store IFR_5.

:SYSTem :TIME

Parameters: <NRf>, <NRf>, <NRf>

hours, minutes, seconds

Valid values: hours: integer. Valid values are 0 to 23. Values outside range are rejected and an error generated.

minutes: integer. Valid values are 0 to 59. Values outside range are rejected and an error generated.

seconds: integer. Valid values are 0 to 59. Values outside range are rejected and an error generated.

Description: Sets the time of the real-time clock. The time is in 24-hour format.

Example:
```
:SYST:TIME, 14, 30, 0
```
Set the time to 2:30 pm.

:SYSTem :TIME?

Parameters: none

Response: <NR1>, <NR1>, <NR1>

hours, minutes, seconds

Returned values: hours: integer. Values are in the range 0 to 23.
minutes: integer. Values are in the range 0 to 59.
seconds: integer. Values are in the range 0 to 59.

Description: Read the time of the real-time clock.

Example: :SYST:TIME?

:SYSTem :TRIGger [:MODE]

Parameters: <CPD>

trigger mode

Valid values: trigger mode: [AUTO | MEASurement]. Values other than those stated are rejected and an error generated.

Description: Set the trigger mode. This is set on a instrument wide basis.

The trigger mode applies to the start of a complete measurement update. So, for instance, if two uncoupled channels are displayed, and the trigger mode is set to one-shot mode, sending a *TRG from the remote controller will cause two measurement acquisition cycles to occur and then stop and wait for another *TRG.

AUTO	The normal "free run" mode. Each measurement update will immediately follow the previous measurement update.
MEASurement	The start of the measurement update will only occur when a GET (group execute trigger) or a *TRG is received from the remote controller.

Example: `:SYST:TRIG AUTO`

Enable auto-trigger mode.

:SYSTem :TRIGger [:MODE]?

Parameters: none

Response: <CRD>

trigger mode

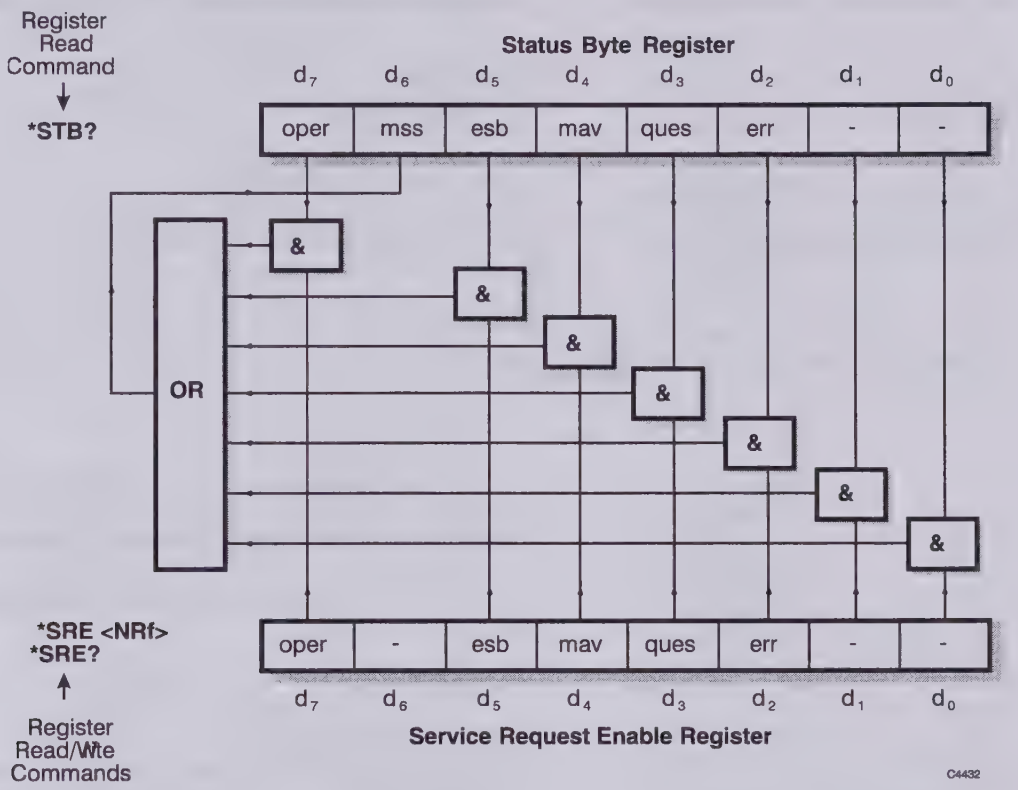
Returned values: trigger mode: [AUTO | MEAS]

Description: Determine the trigger mode.

Example: `:SYST:TRIG?`

Appendix A
GPIB STATUS REPORTING STRUCTURE

Status Byte when read by *STB

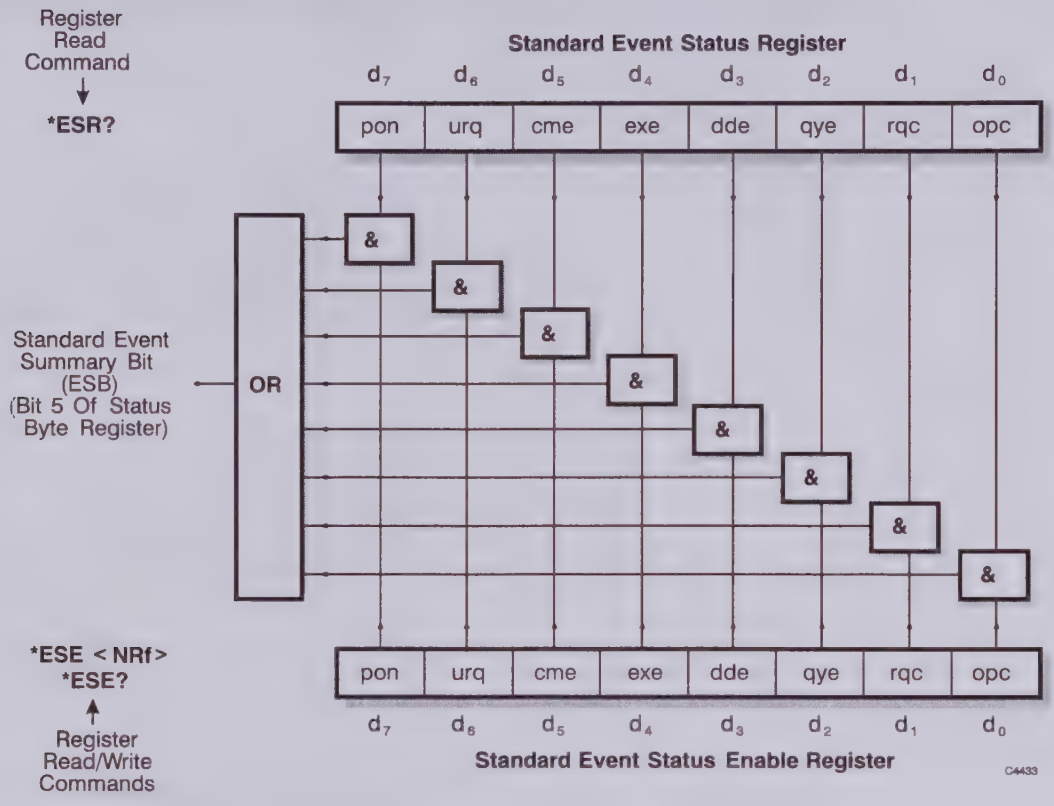


BIT	MNEM	DESCRIPTION
d_0	Not used	
d_1	Not used	
d_2	ERR	Error queue contains at least one error
d_3	QUES	Questionable Status Event Register Summary Bit
d_4	MAV	Message available in output queue (Queue not empty)
d_5	ESB	Standard Event Status Register Summary Bit
d_6	MSS	True when the device has at least one reason for requesting service
d_7	OPER	Operation Status Event Register Summary Bit

Note When read by Serial Poll (rather than *STB?), d_6 contains RQS (Request Service) as defined in IEEE 488.2.

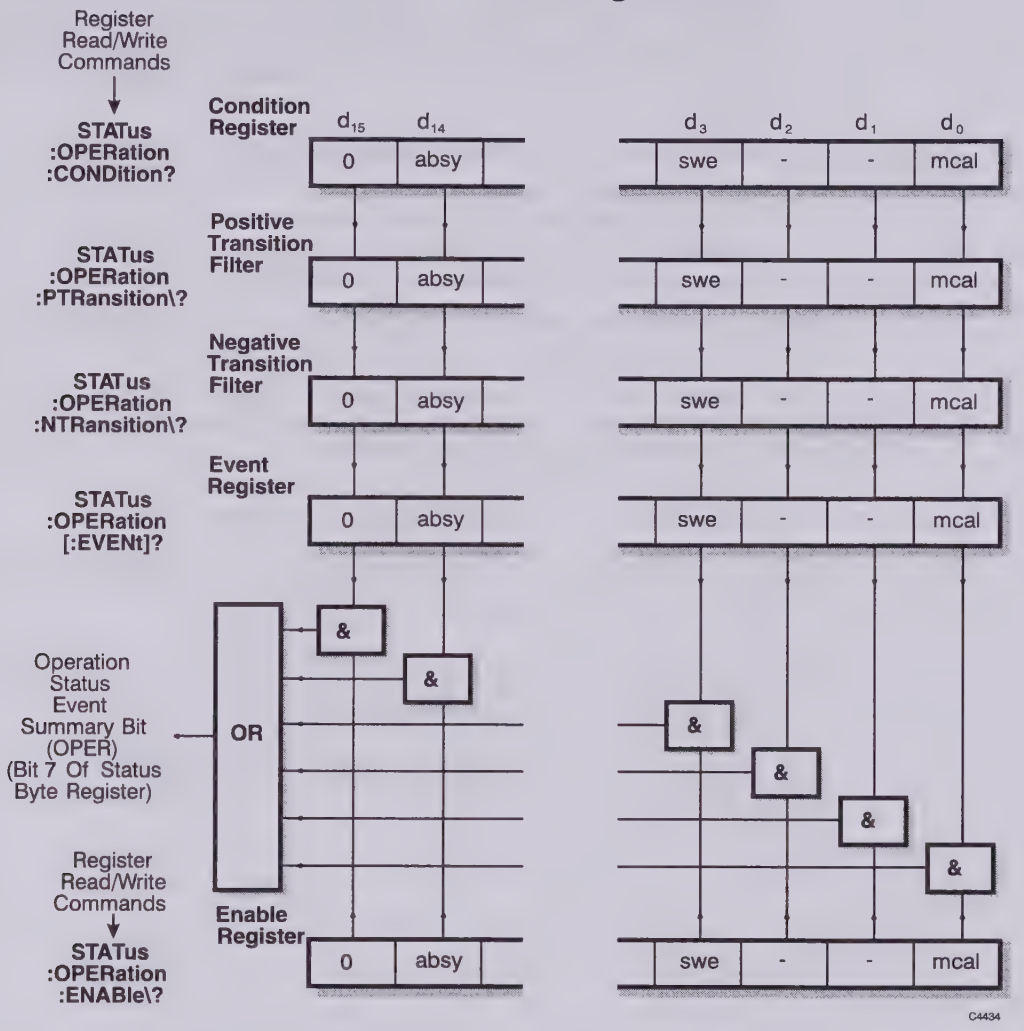
*SRE? always returns 0 for bit d_6.

Standard Event Status Register (as defined in IEEE 488.2)



BIT	MNEM	DESCRIPTION
d_0	OPC	Operation Complete
d_1	RQC	Request Control. The instrument is requesting permission to become the active IEEE 488 controller in charge
d_2	QYE	Query Error
d_3	DDE	Device-Specific Error
d_4	EXE	Execution Error
d_5	CME	Command Error
d_6	URQ	User Request - Not implemented in this instrument
d_7	PON	Power on

Operation Status Condition / Event / Enable Registers



BIT	MNEM	DEFAULT TRANSITION	DESCRIPTION
d_0	MCAL	NEG	Measuring for calibration
d_1			Not Used
d_2			Not Used
d_3	SWE	NEG	Sweeping
d_4			Not Used
d_5	TRIG	POS	Waiting for Trigger
d_6			Not Used
d_7			Not Used
d_8	AV11	NEG	Averaging: Chan 1 Meas 1
d_9	AV12	NEG	Averaging: Chan 1 Meas 2
d_{10}	AV21	NEG	Averaging: Chan 2 Meas 1
d_{11}	AV22	NEG	Averaging: Chan 2 Meas 2
d_{12}			Not Used
d_{13}			Not Used
d_{14}	ABSY	NEG	Application busy
d_{15}			Always zero

Notes

The default transitions listed above are those set at power on and when :SYSTem:PRESet is received. Note that the Operation Status Enable Register is cleared to all zeros at power on and on receipt of :SYSTem:PRESet so it is necessary to enable the appropriate bits before the summary bit in the status byte register will be enabled.

Each transition filter can be set independently giving four states:

Operation Status Event disabled
Operation Status Event set on positive transition in condition register
Operation Status Event set on negative transition in condition register
Operation Status Event set on positive or negative transitions in condition register

MCAL is set whenever sweeps are being performed for a scalar, fault location, or spectrum analyzer calibration. This bit can be used to indicate the completion of a calibration.

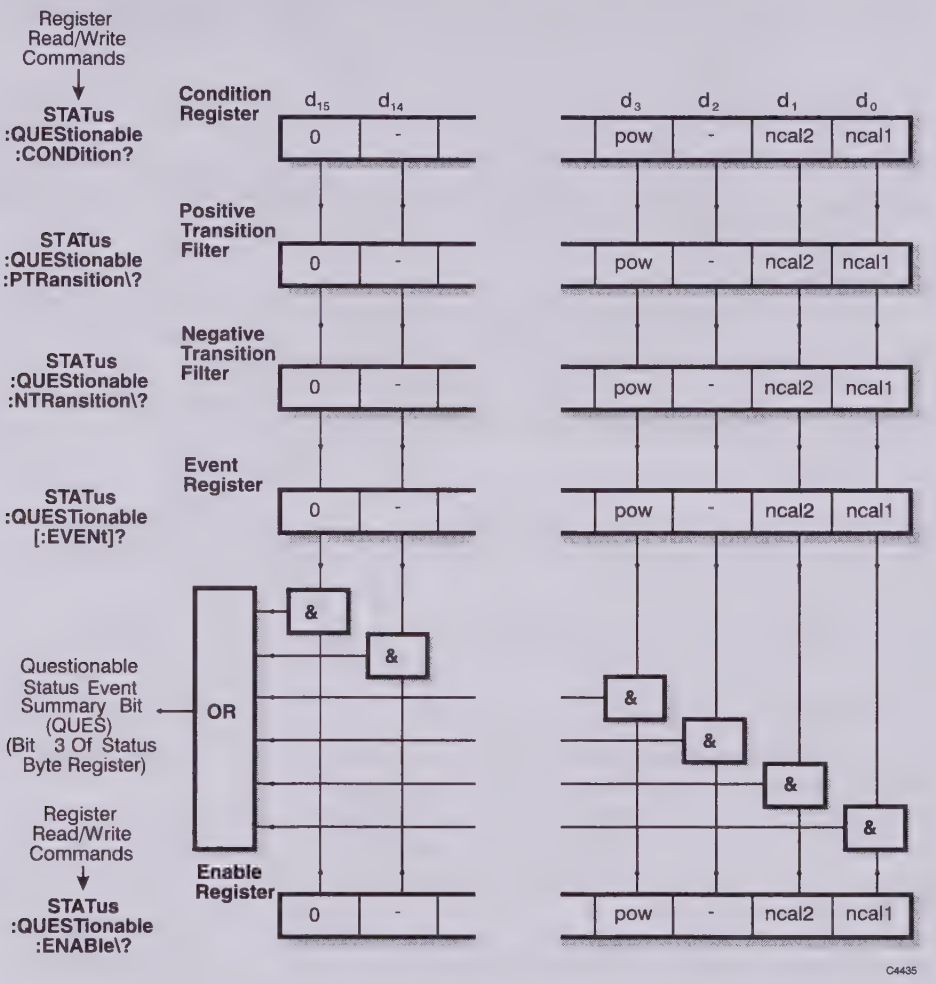
SWE is set at the start of a measurement update and cleared at the end of the measurement update. A measurement update is defined as the time to update all displayed traces (and the audio output if required). If there are two uncoupled channels then those channels will be updated sequentially requiring a sweep for each but the SWEEPING bit will get set at the start of the first channel update and not cleared until the second channel has completed.

TRIG is only used when triggered sweeps have been selected. If the instrument is waiting for a trigger then this bit will be set. It will clear as soon as a trigger has been received. If the instrument is not in triggered mode, this will always be set to 0.

AVnn bits are only used when averaging is on for the measurement. The bit is set high whilst the current averaging has not reached the target value and will clear as soon as the target value is reached. This bit will be set to 0 if averaging is off.

ABSY is only used when an application is running. It is used to indicate that the application has not yet completed an action. The definition of completing an action will be defined in the manual on the application.

Questionable Status Condition / Event / Enable Registers



BIT	MNEM	DEFAULT TRANSITION	DESCRIPTION
d_0	NCAL1	POS	Not calibrated, channel 1
d_1	NCAL2	POS	Not calibrated, channel 2
d_2			Not Used
d_3	POW	POS	Source power unlevelled
d_4	XSTD	POS	External standard not present or standard unlocked
d_5	FREQ	POS	Source unlocked
d_6			Not Used
d_7			Not Used
d_8	CAL	POS	Calibration failure
d_9	LIM11	POS	Out of limits: Chan 1 Meas 1
d_{10}	LIM12	POS	Out of limits: Chan 1 Meas 2
d_{11}	LIM21	POS	Out of limits: Chan 2 Meas 1
d_{12}	LIM22	POS	Out of limits: Chan 2 Meas 2
d_{13}			Not Used
d_{14}			Not Used
d_{15}			Always zero

Notes

The default transitions listed above are those set at power on and when :SYSTem:PRESet is received. Note that the Questionable Status Enable Register is cleared to all zeros at power on and on receipt of :SYSTem:PRESet so it is necessary to enable the appropriate bits before the summary bit in the status byte register will be enabled.

Calibration failure, bit d_8, indicates that calibration data for the internal frequency standard has been corrupted and the instrument is using default calibration data. The bit is reset when the calibration is valid again.

Not calibrated, bits d_0 and d_1, indicate different things depending on the channel type. For spectrum analyzer channels they indicate that the instrument is not displaying a calibrated spectrum analyzer measurement as a result of overriding the normal couplings between parameters. For fault location channels they are set if the trace is not being updated because the calibration is invalid. For scalar channels, these bits will be held at 0.

Appendix B
NUMERIC ENTRY HANDLING

All commands that take a NUMERIC VALUE must also accept the following special forms of numbers:

MINimum	Set the parameter to be the smallest value that can be settable for that parameter, i.e. the value nearest to negative infinity.
MAXimum	Set the parameter to be the largest value that can be settable for that parameter, i.e. the value nearest to positive infinity.
UP	Increment the value by the appropriate step value (defined using the :STEP subsystem).
DOWN	Decrement the value by the appropriate step value (defined using the :STEP subsystem).

Some commands must also accept the special form 'MARKer'. This will cause the active marker domain or response value to be used (as appropriate). The following commands must accept this:

:DISP:CENT MARK	Set the display centre domain value to the active marker domain value.
:DISP:SCAL:RLEV MARK	Set the reference level to the active marker response value.
:DISP:SPAN MARK	Set the display span so that the active marker and delta marker are at the start and end of the display. I.E. set the span to be the difference of the active and delta marker. Gives an error if the delta marker is not displayed.
:DISP:STAR MARK	Set the display start domain value to the active marker domain value.
:DISP:STOP MARK	Set the display stop domain value to the active marker domain value.
MARK:PEAK:THR:VAL MARK	Set the peak left/right threshold value to the active marker response value.
:SAN:RX:CENT MARK	Set the receiver centre frequency to the active marker domain value.
:SAN:RX:SPAN MARK	Set the receiver span from the active marker and delta marker. Gives an error if the delta marker is not displayed.
:SAN:RX:STAR MARK	Set the receiver start frequency to the active marker domain value.
:SAN:RX:STOP MARK	Set the receiver stop frequency to the active marker domain value.
:SOUR:FREQ:CENT MARK	Set the source centre frequency value to the active marker domain value.
:SOUR:FREQ:SPAN MARK	Set the source span so that the active marker and delta marker are at the start and end of the display. I.E. set the span to be the difference of the active and delta marker. Gives an error if the delta marker is not displayed.
:SOUR:FREQ:STAR MARK	Set the source start frequency value to the active marker domain value.

:SOUR:FREQ:STOP MARK	Set the source stop frequency value to the active marker domain value.
:SOUR:POW:STAR MARK	Set the source start power value to the active marker domain value.
:SOUR:POW:STOP MARK	Set the source stop power value to the active marker domain value.
:STEP:FREQ MARK	Set the frequency step value to be the active marker frequency. This is only acceptable when the active channel is a spectrum analyzer.

Appendix C
PARAMETERS NOT AFFECTED BY PRESET

This Appendix lists which parameters are not affected by preset (i.e. *RST). All other parameters are defaulted on receipt of this command.

Serial baud rate	:SYST:SER:BAUD
Serial data bits	:SYST:SER:BITS
Serial stop bits	:SYST:SER:SBIT
Serial parity	:SYST:SER:PAR
Serial flow control	:SYST:SER:FCON
Plotter GPIB address	:HARD:DEV:ADDR
Instrument GPIB address	:SYST:ADDR
Hard copy device	:HARD:DEV
Hard copy port	:HARD:PORT
Display backlight brightness	:DISP:BLIG:VAL
Display backlight state	:DISP:BLIG
Display contrast	:DISP:CONT
Controller source (serial or IEEE488)	:SYST:CONT
Trigger type	:SYST:TRIG
Keyboard layout	:SYST:ISET:KEYB
Language	:SYST:ISET:LANG
Country	:SYST:ISET:COUN
Colour palette	:DISP:CPAL
Distance units (fault location)	:FLOC:UNIT
Application autorun flag	:SYST:APPL:ARUN:STAT
Current application	:SYST:APPL:ARUN
Source power calibration	:SOUR:CAL:SEL:PRI or :SOUR:CAL:SEL:USER
Date format	:SYST:ISET:DATE
Time format	:SYST:ISET:TIME
Field separator	:SYST:ISET:SEP
Decimal point character	:SYST:ISET:DPO

Appendix D
OVERLAPPED COMMANDS

All commands accepted by this instrument are sequential commands apart from those listed below:

:FLOCation:CALibration[:SAVE]

There are two ways of determining when a fault location calibration has completed.

Since the fault location calibration command is an overlapped command, the *OPC, *OPC?, and *WAI commands can be used to determine completion of the calibration. This approach is easier but does not work when the instrument is being triggered using *TRG or device trigger. Note that if a hard copy has also been started, the *OPC etc. will only indicate completion when both the hard copy and the calibration have completed.

If the fault location calibration is being performed using triggered sweeps then it is necessary to determine completion of the calibration using bit 0 of the Operation Status Register.

:HARDcopy:BUILd

:HARDcopy:LIMit:CURRent

:HARDcopy:LIMit[:STORe]

:HARDcopy:OUTPut

:HARDcopy[:PLOT][:ALL]

:HARDcopy[:PLOT]:TRACes

:HARDcopy:SETTings:CURRent

:HARDcopy:SETTings[:STORe]

:HARDcopy:TABLe

:HARDcopy:TEST

The only way of determining the end of the hard copy is by using *OPC, *OPC? or *WAI. The hard copy is deemed to be complete when all of the data has been sent to the hard copy device. If the device has a large buffer this may be before the hard copy has actually been completed.

:SAN:CAL:AMPL

:SAN:CAL:DET

:SAN:CAL:LO

:SAN:CAL:RBF

:SAN:CAL:YIG

There are two ways of determining when a spectrum analyzer calibration has completed.

Since the spectrum analyzer calibration command is an overlapped command, the *OPC, *OPC?, and *WAI commands can be used to determine completion of the calibration. Note that if a hard copy has also been started, the *OPC etc will only indicate completion when both the hard copy and the calibration have completed.

An alternative method to determine completion of the calibration is by using bit 0 (MCAL) of the Operation Status Register.

SAN:NORM:PERF

There are two ways of determining when a normalisation has completed.

Since the normalisation command is an overlapped command, the *OPC, *OPC?, and *WAI commands can be used to determine completion of the normalisation. This approach is easier but does not work when the instrument is being triggered using *TRG or device trigger. Note that if a hardcopy has also been started, the *OPC etc will only indicate completion when both the hardcopy and the normalisation have completed.

If the normalisation is being performed using triggered sweeps then it is necessary to determine completion of the normalisation using bit 3 (SWE) or bit 5 (TRIG) of the operation status register. The normalisation takes only one sweep to perform.

:SCAL:PCAL:OPEN:MERGe[:RLOSs]

:SCAL:PCAL:OPEN:MERGe:SEILoss

:SCAL:PCAL:OPEN[:ONLY][:RLOSs]

:SCAL:PCAL:OPEN[:ONLY]:SEILoss

:SCAL:PCAL:SHORT:MERGe[:RLOSs]

:SCAL:PCAL:SHORT:MERGe:SEILoss

:SCAL:PCAL:SHORT[:ONLY][:RLOSs]

:SCAL:PCAL:SHORT[:ONLY]:SEILoss

:SCAL:PCAL:THRough

There are two ways of determining when a scalar path calibration has completed.

Since the scalar path calibration command is an overlapped command, the *OPC, *OPC?, and *WAI commands can be used to determine completion of the calibration. This approach is easier but does not work when the instrument is being triggered using *TRG or device trigger. Note that if a hard copy has also been started, the *OPC etc. will only indicate completion when both the hard copy and the calibration have completed.

If the scalar path calibration is being performed using triggered sweeps then it is necessary to determine completion of the calibration using bit 0 of the Operation Status Register.

SOUR:CAL:FM

SOUR:CAL:FREQ

SOUR:CAL:POW:BBAN

SOUR:CAL:POW:NBAN

SOUR:CAL:STAN

The only way of determining the end of the source calibration is by using *OPC, *OPC? or *WAI.

Appendix E
EMULATION OF IEEE488.1 ON THE SERIAL INTERFACE

Since the RS-232 interface does not have the extra control lines that the IEEE488.1 interface possesses, it is necessary to emulate the extra functionality. This is done using two mechanisms: the first is the break facility and the second is by sending specific control codes over the RS232 interface which are interpreted to mean specific IEEE488.1 behaviour.

This emulation of IEEE488.1 is only performed when the instrument is using the RS-232 interface to receive commands from an external controller. If the RS-232 is being used to drive a plotter or printer, this does not apply.

Commands from controller to the instrument

There are four messages from the controller to the instrument:

Code sent	ASCII value	Meaning
break signal	N/A	Device clear
^H	8	Device trigger
^R	18	Change to lockout
^X	24	Perform serial poll

The break signal acts as a device clear at any time. Once the device clear has been actioned a reply (&DCL<cr><lf>) is returned. This is necessary because there is no concept of bus holdoff on RS-232.

The emulation codes are accepted at all times except within <ARBITRARY BLOCK PROGRAM DATA> or <STRING PROGRAM DATA> where the data is passed through unchanged.

The instrument will enter remote or remote with lockout states (from local or local with lockout states respectively) on receipt of a byte over the RS-232 interface.

Note that setting the Received Line Signal Detect line (pin 1 of 9-pin RS-232 connector) inactive will force the instrument back to local.

Responses/requests from the instrument to the controller

There are three messages from the instrument to the controller:

Code sent	Meaning
&SRQ<cr><lf>	Request service (asynchronous)
&ddd<cr><lf>	Reply to ^X - STB & RQS sent as three decimal digits (000 - 255)
&DCL<cr><lf>	Acknowledges device clear completion

IFR LTD

SOFTWARE WARRANTY STATEMENT

Definitions

Any software provided by IFR Ltd is supplied subject to the terms of IFR's Software Licence. This is printed on the reverse of our Quotations, will be found with our shrink-wrap Products and is available on request. Software is defined in the Licence as "Licensed Software" and can be further sub-divided as follows:

"Embedded Software" means Licensed Software that forms part of the Equipment supplied by IFR and without which the Equipment cannot function.

"Add-In Application Software" means Licensed Software that may be loaded separately from time to time into the Equipment to improve or modify its functionality.

"Computer Application Software" means Licensed Software supplied to run on a standard PC or workstation.

Warranty

1 IFR certifies that the Licensed Software supplied by IFR will at the time of delivery function substantially in accordance with the applicable Software Product Descriptions, Data Sheets or Product Specifications published by IFR.

2 Unless you have purchased or been provided with extended warranty for Embedded Software the warranty period from date of delivery in respect of each type of Licensed Software is

(i) Embedded Software 12 months

(ii) Add-In Application Software 90 days

(iii) Computer Application Software 90 days

3 If during the appropriate Warranty Period the Licensed Software does not conform substantially to the above Software Product Descriptions, Data Sheets or Product Specifications, IFR will provide

a) In the case of Embedded Software and at IFR's discretion either a fix for the problem or an effective and efficient work around.

b) In the case of Add-In Application Software and Computer Application Software and at IFR's discretion replacement of the software or a fix for the problem or an effective and efficient work around.

IFR does not warrant that the operation of any software will be uninterrupted or error free.

Limitation of Warranty

The above Warranty does not apply to:

(i) Defects resulting from software not supplied by IFR, from unauthorised modification or misuse or from operation outside of the specification.

(ii) Third party produced Proprietary Software which IFR may deliver with its products, in such case the third party Software Licence Agreement including its warranty terms shall apply.

Exclusive Remedies

The remedies offered above are your sole and exclusive remedies and to the extent permitted by applicable law are in lieu of any implied conditions, guarantees or warranties whatsoever and whether statutory or otherwise as to the software all of which are hereby expressly excluded.

Principal Offices

Americas

USA

IFR
10200 West York Street
Wichita, Kansas 67215-8999
USA
Tel: [+1] (316) 522 4981
Fax: [+1] (316) 524 2623
e-mail: info@ifrsys.com

Europe North

UNITED KINGDOM

IFR
Longacres House
Norton Green Road
Stevenage, Hertfordshire SG1 2BA
United Kingdom
Tel: [+44] (1438) 742200
Fax: [+44] (1438) 727601
e-mail: info@ifrsys.com

Europe Central

GERMANY

IFR GmbH
Münchner Str. 87b, 85221 Dachau
Germany
Tel: [+49] (8131) 29260
Fax: [+49] (8131) 2926130
e-mail: info@ifrsys.com

Eastern Europe

GERMANY

IFR GmbH
Robert Bosch Strasse 32
63303 Dreieich-Sprendlingen
Germany
Tel: [+49] (6103) 91180
Fax: [+49] (6103) 911811
e-mail: info@ifrsys.com

Europe South

FRANCE

IFR International SA
18 Rue du Plessis-Briard
"Le Canal"
Courcouronnes
91023 Evry Cedex
France
Tel: [+33] 1 60 79 96 00
Fax: [+33] 1 60 77 69 22
e-mail: info@ifrsys.com

SPAIN

IFR Technologies SA
Europa Empresarial
C/Rozabella, 6
28230 Las Rozas
Madrid
Spain
Tel: [+34] (91) 640 11 34
Fax: [+34] (91) 640 06 40
e-mail: info@ifrsys.com

Asia

CHINA

IFR International Ltd
Room 706/708
7/F Beijing Towercrest Plaza
No 3 Mai Zi Dian Xi Lu
Chao Yang District
Beijing 100016
People's Republic of China
Tel: [+86] (10) 64672716
Fax: [+86] (10) 64672821
e-mail: ifrbj@public.fhnet.cn.net

HONG KONG

IFR International Ltd
Room 702-703
CC Wu Building
302-308 Hennessy Road
Wanchai
Hong Kong
Tel: [+852] 2832 7988
Fax: [+852] 2834 5364
e-mail: info@ifrinternational.com.hk

International Service Centers

All IFR products are supported by the following network of service centers. Details of these establishments may change from time to time. If you require support please contact your local distributor at the relevant address detailed below. If you experience difficulties, please contact the Customer Support Group -CS at our Head Office in the UK.

ARGENTINA
Instruments Sales & Service
VR Marine Electronics SRL
Tucuman 358 Piso 6 "K"
1049 Buenos Aires
Argentina
Tel: (+54) (114) 311 7432
Fax: (+54) (114) 312 2799
e-mail: vrmarine@overnet.com.ar

AUSTRALIA
Instruments & ATE Sales & Service
VICOM Australia Pty. Limited
1064 Centre Road
Oakleigh South
Victoria , 3167
Australia
Tel: (+61) (3) 9563 7844
Fax: (+61) (3) 9579 7255
e-mail: vicom@vicom.com.au

AUSTRIA
Instruments Sales & Service
LB-Electronics GmbH
Döblinger Hauptstrasse 95
A-1190 Wien
Austria
Tel: (+43) (1) 36 030-31
Fax: (+43) (1) 3698443
e-mail: mess@lbe.co.at

AUSTRIA
ATE Sales and Service
IFR GmbH
Münchner Str 87b
85221 Dachau
Germany
Tel: (+49) (8131) 29260
Fax: (+49) (8131) 2926130
e-mail: muick@ibe.co.at

BOTSWANA
See South Africa

BRAZIL
Instruments Sales & Service
Ascom
Rua Braganca Paulista, 158
Vila Cruzeiro
Cep: 04727-000
Sao Paolo- SP
Brazil
Tel: (+55) (11) 56415074
Fax: (+55) (11) 56415081
e-mail: ascom.alix@uol.com.br

BRUNEI
See Hong Kong

BULGARIA
Instruments Sales & Service
GIGA Electronics
45 "Popova Shapka" Str
1505 Sofia
Bulgaria
Tel/Fax: (+359) (2) 732800
e-mail: giga@mobikom.com

CHINA
Instruments Sales & Service
IFR International Ltd
Rm 706/708,
7/F Beijing Towercrest Plaza
No 3 Mai Zi Dian Xi Lu
Chao Yang District
Beijing 100016
People's Republic of China
Tel: (+86) (10) 64672716
Fax: (+86) (10) 64672821
e-mail: ifrbj@public.fhnet.cn.net

COLOMBIA
Instruments Service
RF Labs
Carrera 33 No. 24-55
Interior 102
Santafe de Bogata
Colombia
Tel: (+57) 1 572 4444
Fax: (+57) 1 570 4489
e-mail: rflabs@col1.telecom.com.co

COLOMBIA
Instruments Sales & Service
Rentametric Ltda.
Calle 24 No. 32-67
Santafe de Bogota
Colombia
Tel: (+57) (1) 5709280
Fax: (+57) (1) 2690191
e-mail: rentamet@latino.net.co

CZECH REPUBLIC
Instruments Sales & Service
EMPOS Spol. S.R.O.
Rostislavova 13
140 00 Praha 4
Czech Republic
Tel: (+420) (2) 692 50 80
Fax: (+420) (2) 692 50 84
e-mail: empos@mbox.vol.cz

DENMARK
Instruments & ATE Sales & Service
Stantronic Instruments A/S
Ormhøjgårdvej 16
DK 8700 Horsens
Denmark
Tel: (+45) 75 643366
Fax: (+45) 75 644080
e-mail: info@stantronic.dk

EGYPT
Instruments Sales & Service
Multi-Systems Engineering
28 Ali Amin Street
Nasr City
Cairo
Egypt
Tel: (+20) (2) 2632771
Fax: (+20) (2) 4028057
e-mail: menc@soficom.com.eg

FINLAND
Instruments & ATE Sales & Service
Kaukomarkkinat Oy
Kutojantie 4
P. O. Box 40
SF-02630 Espoo 63
Finland
Tel: (+358) (9) 5211
Fax: (+358) (9) 521 5290
e-mail:
osmo.patteri@kaukomarkkinat.fi

FRANCE
Instruments & ATE Sales & Service
IFR International SA
18 Rue du Plessis-Briard
"Le Canal"
Courcouronnes
91023 Evry Cedex
France
Tel: (+33) 1 60 79 96 00
Fax: (+33) 1 60 77 69 22
e-mail: info@ifrsys.com

GERMANY
Instruments & ATE Sales & Service
IFR GmbH
Münchner Str. 87b
85221 Dachau
Germany
Tel: (+49) (8131) 29260
Fax: (+49) (8131) 2926130
e-mail: info@ifrsys.com

GREECE
Instruments Sales & Service
Theodorou Automation SAICT
3 Aftomedontos Street
116 35 Athens
Greece
Tel: (+30) (1) 7010069
Fax: (+30) (1) 7518829
e-mail: test@theodorou.gr

HONG KONG
Instruments Sales & Service
IFR International Ltd
Room 702-703, CC Wu Building
302-308 Hennessy Road
Wanchai
Hong Kong
Tel: (+852) 2832 7988
Fax: (+852) 2834 5364
e-mail: info@ifrinternational.com.hk

INDIA
Instruments Sales & Service
Blue Star Limited
Sahas, 414/2 Veer Savarkar Marg
Prabhadevi, Bombay 100 025
India
Tel: (+91) (22) 430 6155
Fax: (+91) (22) 430 7078
e-mail:
bn.vijaykumar@bluestar.sprintrpg.ems.vsnl.net.in

INDONESIA
Instruments Sales & Service
PT Panorama Timur Jaya
Jl Paus No. 89 D-E
Rawamangun, Jakarta 13220
Indonesia
Tel: (+62) (21) 4756788
Fax: (+62) (21) 4751688
e-mail: panorama@ub.net.id

IRELAND, REPUBLIC OF
Instruments Sales & Service
Precision Test System
Unit 36, Beechwood Close
Bray
Co. Wicklow
Ireland
Tel: (+353) (1) 286 6777
Fax: (+353) (1) 286 4328
e-mail: lpeare@dataedge.ie

ISRAEL
Instruments Sales & Service
IES Electronics Agencies (1986) Ltd
32 Ben Gurion Street
Ramat Gan 52573
Israel
Tel: (+972) (3) 753 0753
(+972) (3) 753 0700
Fax: (+972) (3) 753 0701
e-mail: gideon@ies.com

ITALY
Instruments Sales & Service
Laser Optronics S.P.A
Via B. Quaranta
57-20139
Milan
Italy
Tel: (+02) 574651
Fax: (+02) 57410127
e-mail:
pampolini@laseroptronic.it

ITALY
Instruments & ATE Sales & Service
Lecroy SA
2 rue du Pre-de-la Fonatire
PO Box 341, CH-1217 Meyrin 1
Genf
Switzerland
Tel: (+41) (22) 7192228
Fax: (+41) (22) 7192230
e-mail:
albert.hanselmann@lecroy.com

JAPAN
Instruments Sales & Service
Kikusui Electronics Corp.
1-1-3, Higashi-Yamata
Tsuzuki-ku
Yokohama City
224-0023
Japan
Tel: (+81) (45) 593 7580
Fax: (+81) (45) 593 7581
e-mail: msuzuki@kikusui.co.jp

JAPAN
Avionics
Kenwood TMI Corp.
1-16-2 Hakusan
Midori-ku
Yokohama-Shi
Kanagawa, 226-8525
Japan
Tel: (+81) (45) 939 7180
Fax: (+81) (45) 939 7089
e-mail: tanaka.s@tmi.kenwood.co.jp

KENYA
See South Africa

MACAU
See Hong Kong

MALAYSIA
Instruments Sales & Service
Mahkota Technologies Sdn Bhd
Wisma Mahkota
Jalan 215-Templer
46050 Petaling Jaya
Selangor
Malaysia
Tel: (+60) (3) 7781 1388
Fax: (+60) (3) 7782 1350
(+60) (3) 7781 1886
e-mail: mktey@tm.net.my

NAMIBIA
See South Africa

NETHERLANDS
Instruments & ATE Sales & Service
T & M Systems
Postbus 645
5000 AP Tilburg
Netherlands
Tel: (+31) (0) 13 463 95 40
Fax: (+31) (0) 13 463 96 63
e-mail: infobni@ifr.nl

NORWAY
Instruments Sales & Service
Nortronicom AS
P.O. Box 33 Manglerud
0612 Oslo
Norway
Tel: (+47) (23) 16 78 90
Fax: (+47) (23) 16 78 99
e-mail: nortronicom@nortronicom.no

OMAN, SULTANATE OF
Instruments Sales & Service
Suhail & Saud Bahwan
PO Box 169
Muscat
Sultanate of Oman
Tel: (+968) 793741
Fax: (+968) 796158
e-mail: pandt@ssbahwan.com

PAKISTAN
Instruments Sales & Service
Intermark Ltd
P.O. Box 6159
Hakimsons Building
19 West Wharf Road
Karachi 74000
Pakistan
Tel: (+92) (21) 2314262
Fax: (+92) (21) 231 04801
e-mail: intermark@cyber.net.pk

PHILIPPINES
Instruments Sales & Service
Sunley Inc
MCPO Box 2850
Makati 1268
Metro Manila
Philippines
Tel: (+63) (2) 818-1216
Fax: (+63) (2) 815-0730
e-mail: sunley@ibm.net

POLAND
Instruments Sales & Service
ZEAP Meratronik SA
Ul. Lopuszanka 117/123
02-047 Warszawa
Poland
Tel: (+48) (22) 659 2540
Fax: (+48) (22) 668 6959
e-mail: mpoltorak@metech.com.pl

PORTUGAL
Instruments Sales & Service
Pinto Basto Electrotecnia e
Máquinas Lda
Av. 24 de Julho, 3 - 1o Esq
1200 LISBOA
Portugal
Tel: (+351) (0) 1 3972041
Fax: (+351) (0) 1 3972595
e-mail: carlos.filipe@pintobasto.pt

QATAR
Instruments Sales & Service
Business Communications Qatar
Ltd
PO Box 150
Doha
Qatar
Tel: (+974) 426161
Fax: (+974) 414100
e-mail: jmtc@qatar.net.qa

ROMANIA
Instruments Sales & Service
Celesta Comexim SRL
17 Splaiul Independentei
BL.101 Sc.C. Apt.41
70501 Bucharest
Romania
Tel: (+40) (1) 3357140
Fax: (+40) (1) 3355031
e-mail: celesta@fx.ro

SAUDI ARABIA
Instruments Sales & Service
Electronic Equipment Marketing
Co
P.O. Box 3750
Riyadh 11481
Kingdom of Saudi Arabia
Tel: (+966) (1) 477 1650
Fax: (+966) (1) 478 5140
e-mail: eemco@nesma.net.sa

SINGAPORE
Instruments & ATE Sales & Service
ROOTS Communications Pte Ltd
150 Kampong Ampat
#06-04 KA Centre
Singapore 368324
Tel: (+65) 283 8500
Fax: (+65) 283 8516
e-mail: tms@roots.com.sg

SLOVENIA
Service
SIQ Slovenian Institute of Quality
and Metrology
Trzaska cesta 2
1000 Ljubljana
Slovenia
Tel: (+386) 61 177 8100
Fax: (+386) 61 177 8444
e-mail: spela.vrbas@siq.si

SOUTH AFRICA
Instruments & ATE Sales & Service
Measuretest cc
PO Box 6301
Dunswart
1508 Gauteng
Republic of South Africa
Tel: (+27) (11) 918 3805/5943/5685
Fax: (+27) (11) 918 5176
e-mail: sales@measuretest.co.za

SPAIN
Instruments & ATE Sales & Service
IFR Technologies SA
Europa Empresarial
C/Rozabella, 6
28230 Las Rozas
Madrid
Spain
Tel: (+34) (91) 640 11 34
Fax: (+34) (91) 640 06 40
e-mail: info@ifrsys.com

SWEDEN
Instruments & ATE Sales & Service
Pronesto AB
Finlandsgatan 18
Box 6014
164 06 Kista
Sweden
Tel: (+46) (8) 444 10 50
Fax: (+46) (8) 751 41 11
e-mail: info@pronesto.se

SWITZERLAND
ATE Sales & Service
Lecroy SA
2 rue du Pre-de-la Fonatire
PO Box 341, CH-1217 Meyrin 1
Genf
Switzerland
Tel: (+41) (22) 7192228
Fax: (+41) (22) 7192230
e-mail:
albert.hanselmann@lecroy.com

TAIWAN
Instruments Sales & Service
HWA Jeat Corp.
6F-3, No. 181 Fu-Hsing N. Road
Taipei 10441
Taiwan
ROC
Tel: (+886) (2) 2719 3216
(+886) (2) 2717 3627
Fax: (+886) (2) 2719 2886
e-mail: hjctwn@ms2.hinet.net

THAILAND
Instruments Sales & Service
Yip in Tsoi & Jacks Ltd
523 Mahaprutharam Road,
Bangrak, Bangkok 10500
Thailand
Tel: (+66) (2) 6390369
Fax: (+66) (2) 2368123
e-mail: jinapim@mozart.inet.co.th

TURKEY
Instruments Sales & Service
Orko Mumessillik
Tunus Caddesi 85/9
Kavaklidere
Ankara 06680
Turkey
Tel: (+90) (312) 426 6047
Fax: (+90) (312) 426 6180
e-mail: orkoas@superonline.com

UAE, ABU DHABI
Instruments Sales & Service
Bahwan Trading Company (LLC)
P O Box 41175
Abu Dhabi, UAE
Tel: (+971) (2) 768909
Fax: (+971) (2) 768907
e-mail: tkmmittc@emirates.net.ae

UNITED KINGDOM
Instruments & ATE Sales & Service
IFR
Longacres House
Norton Green Road
Stevenage
Hertfordshire SG1 2BA
United Kingdom
Tel: (+44) (1438) 742200
Fax: (+44) (1438) 727601
e-mail: info@ifrsys.com

UNITED KINGDOM
Avionics Sales & Service
IFR
Units 14/15
Monks Brook Industrial Park
School Close
Chandlers Ford
Hampshire SO53 4RA
United Kingdom
Tel: (+44) (0) 23 8027 3722
Fax: (+44) (0) 23 8025 4015
e-mail: service@btinternet.com

USA
Instruments Sales & Service
IFR
10200 West York Street
Wichita
Kansas 67215-8999
USA
Tel: (+1) (316) 522 4981
Fax: (+1) (316) 524 2623
e-mail: info@ifrsys.com

VENEZUELA
Instruments Sales & Service
Marcoms De Venezuela C.A.
Edif Exagon, Piso 5, Of. 52D
Av Romulo Gallegos El Marques
Caracas 1070A
Venezuela
Tel: (+58) (2) 242 1110
Fax: (+58) (2) 242 0241
e-mail: rdiaz@marcoms.com.ve

VIETNAM
Instruments Sales & Service
East Asiatic Company Ltd A/S,
Ho Chi Minh City Representative
Office
17 Ton Duc Thang Street, 2nd Floor
Ho Chi Minh City
Vietnam
Tel: (+84) (8) 8290499
(+84) (8) 8290496
Fax: (+84) (8) 8230014
e-mail: nga@eac.com.sg

ZIMBABWE
See South Africa

8/11/00

MP MASTER
MA31409